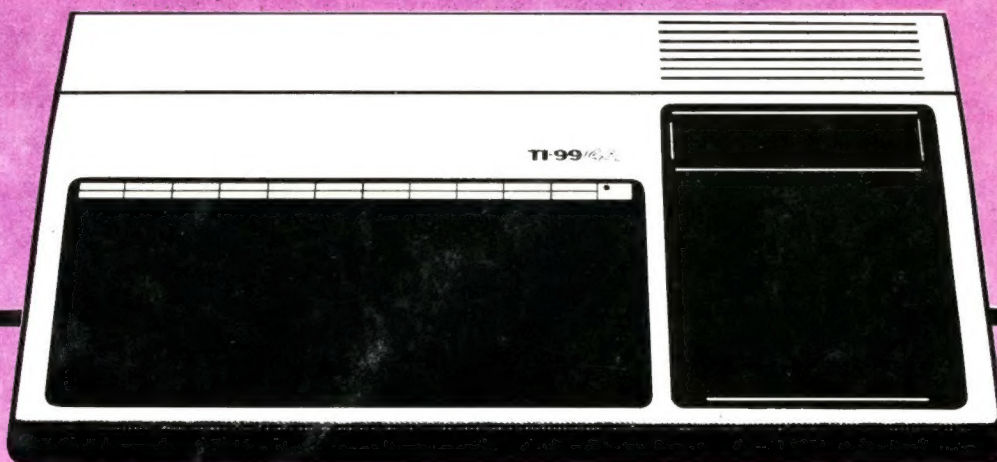


# 99

# MAGAZINE

CASSETTE  
D'ACCOMPAGNEMENT  
DISPONIBLE

*Gestion de fichiers  
transfert image/cassette  
Musique et Mini-mémoire  
Pointeurs et variables en Pascal*



Numéro 9 - Trimestriel -

Juin 1985 - 40 F



### Numéro 1

Musik Maker à l'essai	
Pourquoi acheter un TI-99/4A	
A vous de programmer	
Déballage du TI-99	
Les avantages du Basic étendu	BE
Initiation au Basic	BTI
Jeu du Pendu	BTI
Fabriquez votre joystick	
Appelez moi KEY	BTI/BE
Programmes d'aide à la décision	BE
Jeu de l'Alphabet	BTI
Le club d'utilisateurs TI-Education	
Introduction au LOGO	L
Les tours de Hanoï	BTI
Récupérez les programmes Apple	BTI/BE
Le troisième menuet de Bach	BTI
Le dernier robot	BTI
Cherchez l'ancêtre	BTI
Gestion personnelle de fichiers	
Quand le TI devient un harmonium	BTI
Fichiers en Basic	BE

### Numéro 2

Micro-monde LOGO : les lutins	L
La bataille navale	BTI
Le crocodile savant	
Jeu du Simon	BTI/BE
Jeu des carrés	BTI
L'ordinateur de l'avenir	
TI-Writer à l'essai	
Canon de Purcell	BE
Chicken Helper	BE
Sauvegarde des fichiers et prog.	BTI/BE
Le Mastermind	BTI
Othello	BTI
Astéroïdes	BE
Kermit la grenouille	BE
Le Biorythme	BTI

### Numéro 3

Un programme de calcul	BE
Annonce de Texas Instruments	
Othello à l'essai	
Apprendre Basic en programmant (1)	BTI
Introduction au Pascal	P
Puissance 4	BE
Les modes d'adressage en Assembleur	A
Géographie : la carte de France	BTI
Créez votre touche RESET	
Puzzle	BTI

Awari	
Le sapin de Noël	
PARSEC à l'essai	
LOGO, seulement pour enfant ?	
Le Mastermind en assembleur	
Du nouveau sur les cassettes	
Le magnétophone TI	

### Numéro 4

SOS Hélicoptère	BE
Utilisations de la Mini-mémoire	BTI
Communiqué de Texas Instruments	
Le serveur de restaurant	BTI
"Echecs" à l'essai	
Combat de chars	BTI
Réalisez votre cordon de magnéto	
Apprendre Basic en programmant (2)	BTI
Jeu de mémorisation	BTI
Jack-pot	BE
Les nombres en LOGO	L
Graphisme en haute résolution	BTI + A
Sous-programmes en assembleur	A
Horloge	BE
Un détecteur de lumière pour le TI	
Le Club M.T.I.	
Le système P-UCSD	P
Les lutins du Basic étendu	BE
Le solitaire	BTI
Claustrophobia	BTI
"DISPLAY" et "ACCEPT" avec PRK	BTI
Une erreur dans le manuel MINIMEM	
Réparez votre assembleur ligne par ligne	A

### Numéro 5

Conversions	BTI
Systèmes linéaires	BTI
Structure des disques du P-Système	P
Poèmes et graphismes	BTI
Un désassembleur en assembleur	A
Les tours de Hanoï	BE
Trouvez le bon mot	BTI
Le jeu de la vie	BTI ou A
Programmathèque	
Les opérateurs logiques	BE
Apprendre Basic en programmant (3)	BTI
Bibliographie	
Warangal	BTI
Graphisme en haute résolution	BTI + A
Le trésor de l'Armada	BE
Ecriture littérale d'un nombre	P
Copie d'écran sur imprimante Epson	BTI

BTI	Description du port d'entrées/sorties	
BE	Translet	BE
	Le Club Ordinabis	
L	Le Club Ticazur	
A		

### Numéro 6

	Des chiffres et des lettres	BTI
	A l'heure du Pascal	P
	Kong	BE
BE	Mini-mémoire moins "mini"	
BTI	Edipage	BTI
	Caractères ASCII/page graphique	BTI + A
	Musik	BTI
	Copie d'écran sur Seikosha GP-100	BE
	Nouvelles fonctions avec MMEM	BTI + A
	Le Basic à la loupe	BTI ou BE
	Star Trek	BE
	Ecriture directe dans registres VDP	BTI
	My tailor was rich	BTI
	Le code de la route	BE
	Un double face sur votre TI	

### Numéro 7

	La défasse	BTI
	Le tableur TI-Calc	
	Sauvegarde d'images graphiques	A
	Copie d'écran	A
	Copie d'écran sur Seikosha GP-250	BE
	La compilation séparée	P
	Les bibliothèques	P
	Trouvez votre chemin dans les jeux d'aventure TI	A
	Srand	BTI
	Galaxia	A
	LOGO et la récursivité	L
	Dessinez avec le Basic Etendu	BE
	Accès au mode BitMap en Basic TI	BTI
	Music Box Dancer	BTI
	Des nouvelles du Club Ticazur	
	Routines graphiques pour le BE	BE + A

### Numéro 8

	Bit-Map et Mini-mémoire	BTI + A
	Le train sifflera trois fois...	BE
	Comptes familiaux	BE
	Mission spéciale	BE
	Crayon optique	A
	Le traitement des fichiers en Pascal	P
	Editeur de caractères	BE
	Kaleidoscope	BE
	Opérations	BTI

# Sommaires des numéros 1 à 8 de "99 Magazine"

\* BTI : Basic TI. BE : Basic étendu.

A : Assembleur. P : Pascal. L : Logo.

Tous les numéros sont disponibles.

# **99 Magazine** *numéro 9*

## **Starline**

Georges Goument

25

## **Editorial**

Hervé Thiriez

5

## **Transfert image/ cassette**

22

Gérard Baroni-Jean Marin

## **Pointeurs et variables dynamiques en Pascal**

Gérard Santraille

11

## **Ticrok**

Georges Goument

58

## **Courrier des Lecteurs**

Alexandre Duback

65

## **Gestion de fichiers en Basic étendu et assembleur**

Michel Teyssou

31

## **Un nouveau module graphique**

Henri Mathian

58

## **Musique et Mini- mémoire**

Denise Amrouche

19

## **Petites annonces**

66

## **Module Maximem**

64

## **Pentaxe**

Georges  
Goument

6

# Bon de commande

## Cassettes de jeux

99 A	Mineur, Prêt, Electronique	.....	à	50,00 F	.....
99 B	Yahtzee, Chardef, Division, Régression linéaire	.....	à	50,00 F	.....
99 C	Dames, Nim, Division 2, Isola, Schmoo, Robots	.....	à	50,00 F	.....
99 D	Conjugaisons, Caractérologie, Car Driver, Poker, Cannibales	.....	à	50,00 F	.....
99 E	Mic-Math, Course de chevaux, Poursuite, Jeu de dés, Guerre atomique, Course en ligne droite	.....	à	50,00 F	.....
	Mastermind	.....	à	50,00 F	.....

## Disquettes de jeux

	Mastermind (cf. 99 Magazine n° 3)	.....	à	55,00 F	.....
	Galaxia (cf. 99 Magazine n° 7)	.....	à	55,00 F	.....

## Disquettes utilitaires

	Routines graphiques (cf. 99 Magazine n° 4, 5, 6 et 7)	.....	à	55,00 F	.....
	Désassembleur (cf. 99 Magazine n° 5)	.....	à	55,00 F	.....
	Crayon optique (cf. 99 Magazine n° 8)	.....	à	55,00 F	.....

## Anciens numéros

99 Magazine	1 2 3 4 5 6 7 8 9	.....	à	40,00 F	.....
Cassette d'accompagnement	1 2 3 4 5 6 7 8 9	.....	à	55,00 F	.....

## Abonnements

	Abonnement pour 4 numéros (1 an) à compter du n°	.....	à	135,00 F	.....
	Abonnement aux cassettes d'accompagnement	.....	à	190,00 F	.....

Total TTC .....

Supplément pour port avion hors CEE  
15,00 F par revue et/ou cassette et/ou disquette .....

Montant du règlement joint .....

Nom : .....

Adresse : .....

Envoyer ce bon et votre règlement à :

**Editions MEV - 64, rue des Chantiers - 78000 Versailles**

# Editorial

Ça ne va pas très bien, en ce moment, dans le petit monde de la micro-informatique. Pratiquement tous les constructeurs d'ordinateurs - familiaux ou non - licencient une partie, souvent loin d'être négligeable, de leur personnel. Même IBM, qui n'est pas la plus petite société dans ce domaine, n'échappe pas à cette vague de morosité. La presse spécialisée souffre de cet état de fait, puisque les contrats de publicité se font plus rares, et il faut savoir que les revenus de pratiquement toutes les revues proviennent, pour la plus grande part, de la pub. Dans ce contexte, il est probable que quelques publications spécialisées en micro-informatique seront contraintes à cesser leurs activités.

Heureusement, la politique commerciale et éditoriale des Editions MEV fait que "99 Magazine" n'a pas besoin de publicité pour vivre ; donc, ça va plutôt bien pour nous, même si ce numéro paraît très en retard, cela à cause de problèmes techniques et de personnel. Nous vous prions de bien vouloir nous excuser pour ce décalage, mais je pense que vous serez d'accord avec moi, il vaut mieux un "99" en retard que pas de "99" du tout. Le fait que "99 Magazine" marche bien nous pose quelques problèmes d'intendance ; en effet, nous nous sommes trouvés en rupture de stock pour quelques disquettes et cassettes, en particulier la 6, mais nous espérons que tout sera rentré dans l'ordre lorsque vous lirez ces lignes.

Après ces bonnes paroles, voyons un peu ce que contient ce numéro. Certains d'entre-vous nous ont reproché de ne pas avoir assez mis de programmes en Basic TI dans le numéro 8 ; vous trouverez donc cette fois trois programmes écrits en ce langage ("Starline", "Ticrok" et "Pentaxe") par **Georges Goument**. Vous saurez beaucoup de choses sur les transferts d'informations entre le TI-99/4A et un magnétophone, après avoir lu l'article de **Gérard Baroni** et **Jean Marin**, les deux "piliers" du Club Ticazur, dont il n'est pas inutile de rappeler les coordonnées : Club Ticazur - Gérard Baroni - Villa Hélène - Chemin des Bas-Campons - 06480 La colle-sur-Loup - Tél. : 32.63.16.

**Gérard Santraille** vous communique un maximum d'informations sur les pointeurs et variables dynamiques en Pascal, alors que **Denise Amrouche** vous indique un moyen simple et efficace pour faire cohabiter musique et programmes Basic. **Henri Mathian** vous présente le module "Apesoft", qui est un Basic étendu à vocation graphique distribué en France par "la règle à calcul", qui distribue aussi depuis peu des lecteurs et contrôleurs de disquettes pour TI-99. "99 Magazine" vous présente aussi un tout nouveau module - il n'est pas encore commercialisé - le module "Maximem 48Ko".

Outre les rubriques habituelles "Courrier des Lecteurs", par **Alexandre Duback**, et "Petites annonces" vous trouverez deux programmes de gestion de fichiers écrits par **Michel Teyssou**, l'un écrit en assembleur, l'autre en Basic étendu. La juxtaposition de deux programmes identiques, mais écrits en deux langages diamétralement opposés présente un intérêt pédagogique indéniable.

"99 Magazine" sera à la boutique Sicob (Porte Maillot - Stand M279), nous vous-y attendrons !

*Hervé Thiriez*

Rédacteur en chef - Directeur de la publication : Hervé Thiriez. Rédaction : Jean-Luc Bazanegue, Gérard Santraille. Dessins : Laurent Bidot. Ont collaboré à ce numéro : Denise Amrouche, Gérard Baroni, Jean-Luc Bazanegue, Alexandre Duback, Georges Goument, Jean Marin, Henri Mathian, Gérard Santraille, Michel Teyssou.

Editions MEV - 64/70, rue des Chantiers - 78000 Versailles - Tél. : (3) 951.24.43.

Publicité : consultez les Editions MEV.

Impression : Imprimerie du Lion - Repro-Versailles - 25, rue Michel Lecomte - 75003 Paris - Tél. : 272.96.19.



# Pentaxe

Georges Goument

**C**e jeu de tactique est adapté du morpion Coréen. Nous vous proposons quelques commentaires, afin de compléter les règles du jeu affichées par le programme.

- Il est possible de prendre plusieurs paires de pions adverses en une seule fois. A vous de jouer afin d'obtenir cette possibilité.
- Après une prise, l'adversaire peut rejouer aux mêmes emplacements sans risques, contrant ainsi le jeu de celui qui vient de prendre.
- Pour réaliser un alignement, l'emplacement du dernier pion posé n'a aucune importance ; le programme prend en compte les cinq positions possibles.

Ne vous emmêlez pas les touches en tapant les lignes 339 à 407 ; le déroulement du jeu serait probablement compromis.

Il ne vous reste qu'à trouver un adversaire de valeur pour passer quelques bons moments.

## Déroulement du programme

- 27 à 80 : règles du jeu.
- 81 à 104 : redéfinition des caractères.
- 105 à 108 : variables P1/P2 = prises pions adverses ; G1/G2 = parties gagnées.
- 109 à 151 : dessin du jeu.
- 152 à 173 : changement de joueur. Variables A et B = pions selon joueur, change la couleur d'affichage selon J=0 ou J=1.

174 à 176 : affiche la première coordonnée (ligne=Y) côté gauche ou droit.

177 à 284 : attente d'une touche ; A à Q = coordonnées du pion à jouer. La touche zéro annule la coordonnée Y. La touche <ENTER> annule la partie.

285 - 286 : affiche la seconde coordonnée (colonne=X).

287 à 300 : vérifie que la position jouée est libre. Sert également pour la touche "0" (AN=1).

301 à 305 : le pion est posé sur le jeu ; permute J pour le tour suivant.

306 à 338 : contrôle dans les huit directions possibles, la présence de pions sur quatre intersections maximum.

339 à 407 : selon positions et couleurs des pions détectés, détermine :

- soit une prise (avec retour pour suite du programme) ;
- soit un alignement (partie gagnée, envoi ligne 422) ;
- soit un renvoi ligne 152 pour continuer la partie.

408 à 421 : prise d'une paire de pions. Affichage selon le joueur à droite ou à gauche face à P.

422 à 465 : alignement réalisé ; texte vert ou rouge selon le vainqueur. Affichage identique prise face à G.

466 à 471 : annulation d'une partie.

472 à 478 : nouvelle partie ou arrêt du jeu.

479 à 487 : résultat final (10 parties maximum).

488 à 500 : tirage au sort du premier joueur. Ensuite, donne le jeu au perdant.

## Programme Pentaxe

### Basic TI

```
10 REM *****
11 REM *
12 REM * PENTAXE *
13 REM *
14 REM * programme *
15 REM * en *
16 REM * Basic TI *
17 REM *
18 REM * copyright *
19 REM *
20 REM * 99 Magazine *
21 REM *
22 REM * & *
23 REM *
24 REM * G. Goument *
25 REM *
26 REM *****
27 CALL CLEAR
28 FOR I=1 TO 12
29 CALL COLOR(I,1,1)
30 NEXT I
31 CALL CHAR(33,"FFFFFF
FFFFFFFFFFFF")
32 PRINT TAB(9);"*****
*****":TAB(9);"!
!!!!!!!!!!":TAB(9);"
*!PENTAXE!*":TAB(9);
"!!!!!!!!!!!"
33 PRINT TAB(9);"*****
*****":TAB(11);"j
eu de":TAB(10);"ta
ctique":TAB(9);"2
joueurs"::::
34 PRINT "<c> g goumen
t decembre 1984"::::
35 CALL COLOR(1,12,1)
36 CALL SCREEN(8)
37 CALL COLOR(2,10,14)
38 FOR I=3 TO 12
39 CALL COLOR(I,13,1)
40 NEXT I
41 FOR I=5 TO 8
42 CALL COLOR(I,7,12)
43 NEXT I
44 RESTORE 48
45 FOR I=1 TO 53
46 READ D,N
```

7

139 CALL VCHAR(5,I,Q,17)	191 IF C=1 THEN 194	253 GOTO 174
140 NEXT I	192 Y=6	254 X=19
141 CALL COLOR(1,2,2)	193 GOTO 174	255 GOTO 285
142 CALL COLOR(2,13,15)	194 X=9	256 IF T<>77 THEN 262
143 CALL COLOR(3,8,1)	195 GOTO 285	257 IF C=1 THEN 260
144 CALL COLOR(4,8,1)	196 IF T<>67 THEN 202	258 Y=17
145 CALL COLOR(6,5,15)	197 IF C=1 THEN 200	259 GOTO 174
146 CALL COLOR(7,5,15)	198 Y=7	260 X=20
147 CALL COLOR(8,9,15)	199 GOTO 174	261 GOTO 285
148 FOR I=9 TO 11	200 X=10	262 IF T<>78 THEN 268
149 CALL COLOR(I,12,1)	201 GOTO 285	263 IF C=1 THEN 266
150 NEXT I	202 IF T<>68 THEN 208	264 Y=18
151 GOTO 488	203 IF C=1 THEN 206	265 GOTO 174
152 IF J=1 THEN 161	204 Y=8	266 X=21
153 M=65	205 GOTO 174	267 GOTO 285
154 Z=5	206 X=11	268 IF T<>79 THEN 274
155 A=40	207 GOTO 285	269 IF C=1 THEN 272
156 B=88	208 IF T<>69 THEN 214	270 Y=19
157 CO=3	209 IF C=1 THEN 212	271 GOTO 174
158 CALL HCHAR(2,28,32,2)	210 Y=9	272 X=22
159 CALL HCHAR(2,3,41,2)	211 GOTO 174	273 GOTO 285
160 GOTO 168	212 X=12	274 IF T<>80 THEN 280
161 M=64	213 GOTO 285	275 IF C=1 THEN 278
162 Z=27	214 IF T<>70 THEN 220	276 Y=20
163 A=88	215 IF C=1 THEN 218	277 GOTO 174
164 B=40	216 Y=10	278 X=23
165 CO=10	217 GOTO 174	279 GOTO 285
166 CALL HCHAR(2,3,32,2)	218 X=13	280 IF T<>81 THEN 177
167 CALL HCHAR(2,28,89,2)	219 GOTO 285	281 IF C=1 THEN 284
168 CALL COLOR(5,CO,16)	220 IF T<>71 THEN 226	282 Y=21
169 CALL SOUND(3,1966,11)	221 IF C=1 THEN 224	283 GOTO 174
170 CALL SOUND(10,1949,11)	222 Y=11	284 X=24
171 Y=0	223 GOTO 174	285 CALL HCHAR(24,X,66)
172 X=0	224 X=14	286 C=0
173 GOTO 177	225 GOTO 285	287 CALL SOUND(1,770,10)
174 CALL HCHAR(Y,Z,M)	226 IF T<>72 THEN 232	288 CALL HCHAR(Y,Z,32)
175 CALL SOUND(1,770,10)	227 IF C=1 THEN 230	289 IF AN=1 THEN 295
176 C=1	228 Y=12	290 CALL HCHAR(24,X,32)
177 CALL KEY(3,T,S)	229 GOTO 174	291 CALL GCHAR(Y,X,F)
178 IF S=0 THEN 177	230 X=15	292 IF F=40 THEN 295
179 IF T<>48 THEN 182	231 GOTO 285	293 IF F=88 THEN 295
180 AN=1	232 IF T<>73 THEN 238	294 GOTO 301
181 GOTO 288	233 IF C=1 THEN 236	295 FOR I=1 TO 5
182 IF T<>13 THEN 184	234 Y=13	296 CALL SOUND(10,110,0)
183 GOTO 466	235 GOTO 174	297 NEXT I
184 IF T<>65 THEN 190	236 X=16	298 AN=0
185 IF C=1 THEN 188	237 GOTO 285	299 C=0
186 Y=5	238 IF T<>74 THEN 244	300 GOTO 177
187 GOTO 174	239 IF C=1 THEN 242	301 CALL HCHAR(Y,X,A)
188 X=8	240 Y=14	302 IF J=1 THEN 305
189 GOTO 285	241 GOTO 174	303 J=1
190 IF T<>66 THEN 196	242 X=17	304 GOTO 306
	243 GOTO 285	305 J=0
	244 IF T<>75 THEN 250	306 CALL GCHAR(Y-1,X-1,H1)
	245 IF C=1 THEN 248	307 CALL GCHAR(Y-2,X-2,H2)
	246 Y=15	308 CALL GCHAR(Y-3,X-3)
	247 GOTO 174	
	248 X=18	
	249 GOTO 285	
	250 IF T<>76 THEN 256	
	251 IF C=1 THEN 254	
	252 Y=16	



,H3)	340 D=Y-1	374 GOSUB 408
309 CALL GCHAR(Y-4,X-4	341 E=X-1	375 IF (H13=A)*(H14=A)
,H4)	342 F=Y-2	*(H15=A)*(H16= A)T
310 CALL GCHAR(Y-1,X,H	343 G=X-2	HEN 422
5)	344 GOSUB 408	376 IF (H15=A)*(H14=A)
311 CALL GCHAR(Y-2,X,H	345 IF (H1=A)*(H2=A)*(	*(H13=A)*(H29= A)T
6)	H3=A)*(H4=A) THEN	HEN 422
312 CALL GCHAR(Y-3,X,H	422	377 IF (H14=A)*(H13=A)
7)	346 IF (H3=A)*(H2=A)*(	*(H29=A)*(H30= A)T
313 CALL GCHAR(Y-4,X,H	H1=A)*(H17=A) THEN	HEN 422
8)	422	378 IF (H13=A)*(H29=A)
314 CALL GCHAR(Y-1,X+1	347 IF (H2=A)*(H1=A)*(	*(H30=A)*(H31= A)T
,H9)	H17=A)*(H18=A) THE	HEN 422
315 CALL GCHAR(Y-2,X+2	N 422	379 IF (H17<>B)+(H18<>
,H10)	348 IF (H1=A)*(H17=A)*	B)+(H19<>A) THEN 3
316 CALL GCHAR(Y-3,X+3	(H18=A)*(H19= A)TH	85
,H11)	EN 422	380 D=Y+1
317 CALL GCHAR(Y-4,X+4	349 IF (H5<>B)+(H6<>B)	381 E=X+1
,H12)	+(H7<>A) THEN 355	382 F=Y+2
318 CALL GCHAR(Y,X+1,H	350 D=Y-1	383 G=X+2
13)	351 E=X	384 GOSUB 408
319 CALL GCHAR(Y,X+2,H	352 F=Y-2	385 IF (H17=A)*(H18=A)
14)	353 G=X	*(H19=A)*(H20 =A)T
320 CALL GCHAR(Y,X+3,H	354 GOSUB 408	HEN 422
15)	355 IF (H5=A)*(H6=A)*(	386 IF (H21<>B)+(H22<>
321 CALL GCHAR(Y,X+4,H	H7=A)*(H8=A) THEN	B)+(H23<>A) THEN 3
16)	422	92
322 IF Y>20 THEN 335	356 IF (H7=A)*(H6=A)*(	387 D=Y+1
323 CALL GCHAR(Y+1,X+1	H5=A)*(H21=A) THEN	388 E=X
,H17)	422	389 F=Y+2
324 CALL GCHAR(Y+2,X+2	357 IF (H6=A)*(H5=A)*(	390 G=X
,H18)	H21=A)*(H22=A) THE	391 GOSUB 408
325 CALL GCHAR(Y+3,X+3	N 422	392 IF (H21=A)*(H22=A)
,H19)	358 IF (H5=A)*(H21=A)*	*(H23=A)*(H24 =A)T
326 CALL GCHAR(Y+4,X+4	(H22=A)*(H23= A)TH	HEN 422
,H20)	EN 422	393 IF (H25<>B)+(H26<>
327 CALL GCHAR(Y+1,X,H	359 IF (H9<>B)+(H10<>B	B)+(H27<>A) THEN 3
21)	)+(H11<>A) THEN 365	99
328 CALL GCHAR(Y+2,X,H	360 D=Y-1	394 D=Y+1
22)	361 E=X+1	395 E=X-1
329 CALL GCHAR(Y+3,X,H	362 F=Y-2	396 F=Y+2
23)	363 G=X+2	397 G=X-2
330 CALL GCHAR(Y+4,X,H	364 GOSUB 408	398 GOSUB 408
24)	365 IF (H9=A)*(H10=A)*	399 IF (H25=A)*(H26=A)
331 CALL GCHAR(Y+1,X-1	(H11=A)*(H12= A)TH	*(H27=A)*(H28 =A)T
,H25)	EN 422	HEN 422
332 CALL GCHAR(Y+2,X-2	366 IF (H11=A)*(H10=A)	400 IF (H29<>B)+(H30<>
,H26)	*(H9=A)*(H25= A)TH	B)+(H31<>A) THEN 4
333 CALL GCHAR(Y+3,X-3	EN 422	06
,H27)	367 IF (H10=A)*(H9=A)*	401 D=Y
334 CALL GCHAR(Y+4,X-4	(H25=A)*(H26= A)TH	402 E=X-1
,H28)	EN 422	403 F=Y
335 CALL GCHAR(Y,X-1,H	368 IF (H9=A)*(H25=A)*	404 G=X-2
29)	(H26=A)*(H27= A)TH	405 GOSUB 408
336 CALL GCHAR(Y,X-2,H	EN 422	406 IF (H29=A)*(H30=A)
30)	369 IF (H13<>B)+(H14<>	*(H31=A)*(H32 =A)T
337 CALL GCHAR(Y,X-3,H	B)+(H15<>A) THEN 3	HEN 422
31)	75	407 GOTO 152
338 CALL GCHAR(Y,X-4,H	370 D=Y	408 CALL SOUND(300,-4,
32)	371 E=X+1	0)
339 IF (H1<>B)+(H2<>B)	372 F=Y	409 CALL HCHAR(F,G,Q)
+(H3<>A) THEN 345	373 G=X+2	410 CALL HCHAR(D,E,Q)

```

411 IF J=1 THEN 417
412 P2=P2+1
413 CALL SOUND(5,1200,
0)
414 CALL HCHAR(6,29,P2
)
415 IF P2=53 THEN 422
416 RETURN
417 P1=P1+1
418 CALL SOUND(5,1200,
0)
419 CALL HCHAR(6,3,P1)
420 IF P1=53 THEN 422
421 RETURN
422 IF J=1 THEN 428
423 P$=" rouges "
424 CO=9
425 G2=G2+1
426 IF G2>57 THEN 479
427 GOTO 432
428 P$=" verts "
429 CO=13
430 G1=G1+1
431 IF G1>57 THEN 479
432 CALL SCREEN(16)
433 FOR I=9 TO 12
434 CALL COLOR(I,CO,1)
435 NEXT I
436 FOR REP=1 TO 4
437 CALL SCREEN(2)
438 CALL SOUND(-50,262
,10,196,15, 167,20
)
439 CALL SCREEN(5)
440 CALL SOUND(-50,330
,10,247,15, 208,20
)
441 CALL SCREEN(9)
442 CALL SOUND(-50,247
,10,185,15, 156,20
)
443 CALL SCREEN(12)
444 CALL SOUND(-50,220
,10,165,15, 139,20
)
445 CALL SCREEN(3)
446 CALL SOUND(-50,262
,10,196,15, 167,20
)
447 CALL SCREEN(15)
448 CALL SOUND(-50,294
,10,220,15, 185,20
)
449 NEXT REP
450 CALL SCREEN(2)
451 M$="les"&P$&"sont
vainqueurs"
452 GOSUB 457
453 CALL HCHAR(24,3,32
,28)
454 M$="pour rejouer a
ppuyez sur o"

```

```

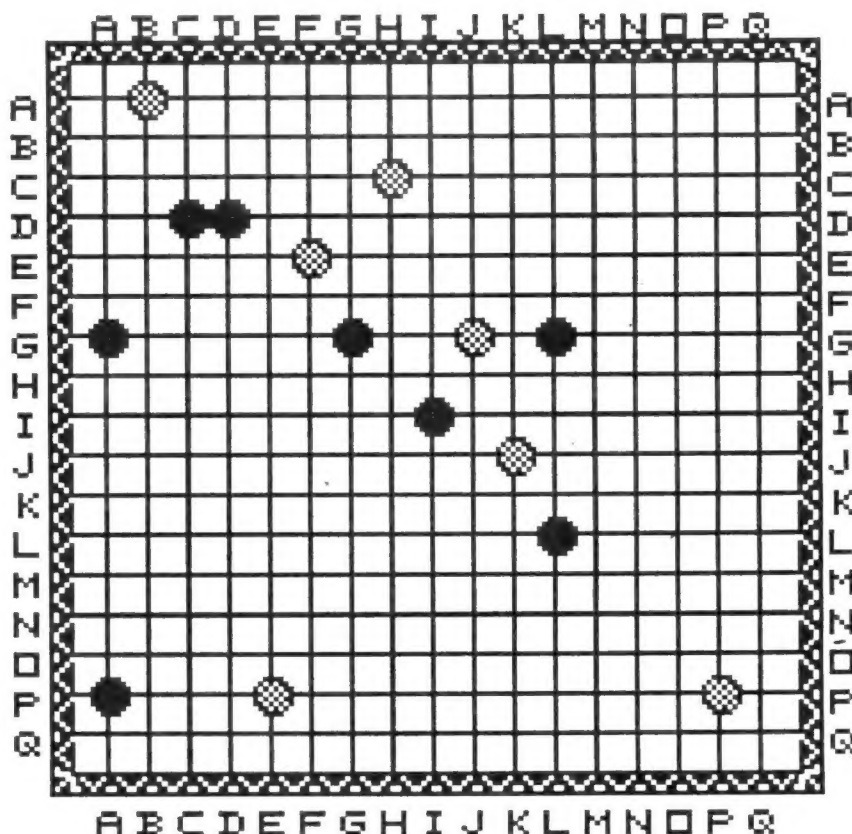
455 GOSUB 457
456 GOTO 472
457 FOR I=1 TO 4
458 CALL SOUND(10,1946
,11)
459 NEXT I
460 FOR I=1 TO LEN(M$)
461 CALL HCHAR(24,2+I,
ASC(SEG$(M$, I,1))
)
462 NEXT I
463 FOR I=1 TO 100
464 NEXT I
465 RETURN
466 FOR I=1 TO 12
467 CALL COLOR(I,10,1)
468 NEXT I
469 M$="partie annulee
"
470 GOSUB 457
471 GOSUB 453
472 CALL KEY(3,T,E)
473 IF E=0 THEN 472
474 IF T<>79 THEN 479
475 CALL HCHAR(24,3,32
,28)
476 P1=48
477 P2=48

```

```

478 GOTO 134
479 CALL CLEAR
480 IF G1=G2 THEN 484
481 IF G1>G2 THEN 486
482 PRINT "les pions r
ouges gagnent"
483 GOTO 487
484 PRINT "partie null
e"
485 GOTO 487
486 PRINT "les pions v
erts gagnent"
487 END
488 IF RE=1 THEN 491
489 IF J=1 THEN 496
490 IF J=0 THEN 493
491 J=INT((10-1+1)*RND
)+1
492 IF J>5 THEN 496
493 J=1
494 A=40
495 GOTO 498
496 J=0
497 A=88
498 CALL HCHAR(13,16,A
)
499 RE=0
500 GOTO 152

```



# Pointeurs et variables dynamiques en Pascal

Gérard Santraille

**N**ous avons vu, lors du précédent numéro, comment structurer des données en fichiers. Les divers types de fichiers manipulables par le PASCAL sont très riches et très nombreux, mais les tableaux, les articles ou les ensembles ont une taille fixée de façon définitive lors de l'écriture du programme. Avec le type **pointeur**, que nous allons examiner aujourd'hui, il est possible de construire des structures de données internes (c'est-à-dire en mémoire, sans accès aux divers dispositifs de stockage de masse comme les lecteurs de disquettes) de taille indéterminée.

L'avantage par rapport au tableau est immédiat. Pour pouvoir manipuler les éléments d'un tableau, il est nécessaire de déclarer sa taille au début du programme. Si l'on connaît le nombre exact de ses éléments on ne "gaspille" pas inutilement la mémoire (celle du TI est précieuse car rare) ; mais dans la grande majorité des applications on ne sait pas au début de l'exécution le nombre d'éléments qui seront manipulés. Bien souvent, pour s'affranchir de ce problème, on "surdimensionne" le tableau réservant ainsi de la mémoire non nécessairement utilisée.

Les pointeurs mettent en jeu des structures extensibles qui sont composées d'éléments créés **dynamiquement** lors de l'exécution. Ces éléments ne sont pas directement référencés par un indice, comme dans un tableau,

mais sont désignés (pointés) par une variable pointeur.

## Variables statiques et variables dynamiques

Une variable statique est déclarée dans un programme ou dans une procédure de façon univoque. Lors de l'exécution le programme affecte un emplacement mémoire à cette variable (la quantité de mémoire utilisée dépend du type de cette dernière). Que cette variable soit effectivement utilisée ou non son emplacement est définitif et ne peut par conséquent être ré-affecté pour une autre grandeur. On ne peut donc pas, à l'aide de variables statiques définir des structures de taille indéterminée.

Les variables dynamiques sont créées en cours d'exécution par l'instruction **new** dont nous verrons plus loin la syntaxe exacte. Il n'est pas possible lors de l'écriture du programme de leur donner un nom. Ces variables n'apparaissent donc dans aucune déclaration du programme.

### Variable pointeur

Les variables dynamiques sont désignées (pointées) par

des variables (statiques ou dynamiques) de type pointeur. Le type pointeur n'a de sens que par rapport à un type déjà défini. Le point commun entre une variable pointeur et l'ensemble des variables qui peuvent être désignées par ce pointeur est le type des variables pointées. Une variable pointeur ne peut repérer que des variables dynamiques de même type.

*Relisez bien ce paragraphe, il est fondamental.*

La désignation d'un type pointeur se fait par le symbole "**^**" suivi du type des variables qu'il va pointer.

Ainsi, dans l'exemple suivant :

*Voir programme 1*

### Programme 1

type

PERSONNE = record

nom : string[20];  
prenom : string[15];  
tel : string[15];  
adr1 : string[20];  
adr2 : string[20];  
end;

PTPERS = ^PERSONNE

var

P1,P2 : PTPERS;



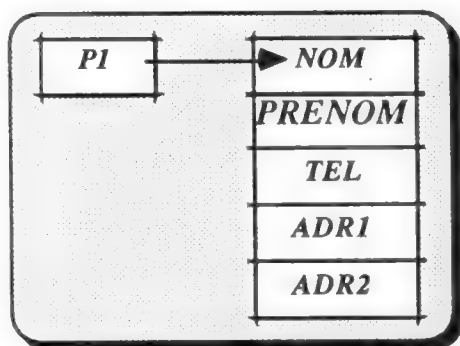
PERSONNE est un type article quelconque composé de CINQ enregistrements (NOM, PRE-NOM, TEL, ADR1, ADR2) PTPERS est le type des variables pointeurs des objets de type PERSONNE.

P1 et P2 représentent des variables pointeurs (que nous appellerons simplement pointeurs) de type PTPERS.

### Remarques

- P1 et P2 ne sont pas initialisés par cette déclaration.
- Aucune variable de type PERSONNE n'est créée par la déclaration de P1 ou P2.

Lorsque, lors de l'exécution, des variables de type PERSONNE seront créées, elles pourront être manipulées au moyen des variables P1 et P2.



## Désignation des variables pointées

Une variable pointée est désignée de la même façon que le tampon ou la fenêtre de fichier (voir 99 Magazine numéro 8).

Ainsi, si P1 est un pointeur sur le type PERSONNE, **P1^** est une variable (dynamique) de type personne. Cette variable ne doit en aucun cas être déclarée.

P1^ est une variable à part entière de type PERSONNE dont on peut très classiquement manipuler les divers champs selon la syntaxe **P1^.champ**

P1^.nom représente donc le champ nom (de type string[20]) de la variable P1^ de type PERSONNE.

## Opérations sur les pointeurs

Nous avons déjà vu que la déclaration P1,P2 : PTPERS pour reprendre l'exemple précédent n'initialisait en aucune façon P1 ou P2.

Pour activer un pointeur, on utilise l'instruction new (pointeur).

Ainsi, dans l'exemple suivant :

```
type toto      = record
.....
.....
end;
```

```
var P : ^toto;
```

```
begin
new(P)
.....
```

la commande new(P) alloue dans la mémoire libre une variable de type toto, P pointe sur cette variable qui sera désignée par P^.

### Remarque :

si toto est un type article avec variantes comme :

Voir programme 2

il convient de déclarer la valeur de toutes les variantes lors du new.

Le type toto possède une variante : selon la valeur du champ rel l'article comporte deux champs supplémentaires (societe et position) ou rien du tout.

Ainsi, new(P,prof) réservera en mémoire une variable dynamique de type toto avec la variante prof. Notons au passage que la taille effectivement réservée dépend des variantes sélectionnées et n'est pas la plus grande de ces dernières comme c'est le cas pour une allocation statique (d'où perte de place).

Les opérations possibles sur les pointeurs sont au nombre de trois.

- affectation
  - 1 - égalité de deux pointeurs de même type
  - 2 - pointeur := nil; le pointeur ne repère alors aucune variable.
- logique
  - 3 - on peut tester l'égalité ou l'inégalité de deux pointeurs.

## Premier exemple d'utilisation des pointeurs

Supposons que l'on dispose d'un fichier d'adresses recensant toutes nos relations. Chaque personne est définie par un article dont l'un des champs est une variable booléenne (vraie ou

### Programme 2

```
type  relation  = (prof,perso);
      toto      = record
                                nom      : string[20];
                                prenom   : string[15];

                                case rel : relation of
      prof      : (societe : string[12];position : string[10]);

      perso     : ();
```

fausse) décrivant une caractéristique donnée. Si nous désirons, pour un traitement ultérieur, charger en mémoire toutes les personnes dont ce champ est vrai nous disposons de plusieurs techniques.

### 1 - sans faire appel aux pointeurs

Si **personne** est le type de l'article et **caract** est le nom de la variable booléenne on peut lire tous les enregistrements du fichier et charger dans un tableau tous les enregistrements dont **caract** est vrai.

Comme on ne connaît pas à priori le nombre de personnes susceptibles d'avoir le champ **caract** vrai, il convient de surdimensionner le tableau destinataire.

Le programme pourrait être :

*Voir programme 3*

### 2-En utilisant les pointeurs

Au lieu de surdimensionner à 100 le tableau destinataire des éléments de type **personne** on crée un tableau de pointeurs (ce qui prend beaucoup moins de place)

*Voir programme 4*

Attention à ne pas confondre un tampon de fichier (**fichier^**) avec une variable pointée (**tableau[i]^**).

Dans le premier programme chaque personne ayant le champ **caract** vrai se trouve physiquement dans **tableau**. Dans le second, **tableau[i]** est un pointeur et **tableau[i]^** est la variable pointée (de type **personne**) correspondante.

L'espace mémoire ainsi sauvegardé est très important. En effet, la taille du tableau des pointeurs additionnée à celle des variables dynamiques de type **personne** est bien inférieure à la taille d'un tableau de type **personne**.

### Remarque

Dans cet exemple nous avons utilisé un ensemble de variables statiques (le tableau des pointeurs) pour repérer un ensemble de variables dynamiques (les personnes ayant le champ **caract** vrai). En effet le tableau de pointeurs a été déclaré comme tel avant l'exécution du programme. Nous disposons d'autant de pointeurs que de variables pointées.

## Programme 3

```
program NOPOINT;
type
    personne = record
    .....
    caract : boolean;
    .....
end;

var
    i           : integer;
    fichier     : file of personne;
    tableau     : array [1..100] of personne;

begin
    reset(fichier);
    i:=0;
    while not eof(fichier) do
        if fichier^.caract then (* sous entendu vrai *)
        begin
            i:=i+1;
            tableau[i]:=fichier^
        end;
    end;
    get(fichier)
end
end
```

## Programme 4

```
program POINT;
type
    personne = record
    .....
    caract : boolean;
    .....
end;
    pointeur = ^personne;

var
    i           : integer;
    fichier     : file of personne;
    tableau     : array [1..100] of pointeur;

begin
    reset(fichier);
    i:=0;
    while not eof(fichier) do
        if fichier^.caract then
        begin
            i:=i+1;
            new(tableau[i]);
            tableau[i]^:=fichier^
        end;
    end;
    get(fichier)
end
end.
```

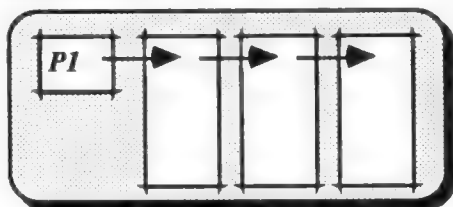
L'avantage de cette technique est que chaque variable dynamique est accessible **directement** puisqu'on dispose, pour chacune d'entre elles, de son pointeur (qui rappelons le est une variable statique). Son principal inconvénient est le même que pour le premier exemple : il est nécessaire de réserver tout un tableau (celui des pointeurs) qui ne sera pas nécessairement utilisé en totalité. Pour pallier cet inconvénient on peut imaginer de pointer les variables dynamiques par des pointeurs eux-mêmes créés dynamiquement.

Ces ensembles de données où les variables pointées se déduisent par récurrence s'appellent **structures chaînées** ou **listes**.

## Structures de données récursives.

On peut imaginer un ensemble de variables dynamiques ayant chacune un champ constitué par un pointeur référençant la variable suivante.

On peut schématiser cette structure selon



où P1 est un pointeur qui désigne une première variable. Cette variable contient un pointeur qui permet d'accéder à la seconde, etc...

De telles variables doivent donc avoir la structure suivante :

```
type TOTO = record
    .....
    suivant : ^TOTO;
    .....
end;
```

Un champ de l'article est donc un pointeur du type **auquel** il appartient.

En PASCAL cette écriture est normalement interdite puisqu'un article ne peut contenir un champ de son propre type. Une exception est faite pour les pointeurs. On doit d'abord définir le type des pointeurs sur les éléments de la liste, puis le type des éléments de la liste.

L'écriture correcte est donc:

```
type PTTOTO = ^TOTO
TOTO = record
    .....
    suivant : PTTOTO;
    .....
end;
```

Une liste récursive simple dont chaque élément possède le pointeur du suivant n'admet qu'un seul mode d'accès. Une fois la liste définie on ne peut remonter au premier élément qu'en parcourant tous les autres.

Il s'agit d'une gestion d'accès de type **LIFO** (de l'anglais Last In - First Out, littéralement dernier entré - premier sorti) : on retombe sur l'éternel problème de la pile d'assiettes.

Pour mieux comprendre ce qui se passe exécutons le petit programme suivant:

*Voir programme 5*

Ce programme permet de saisir cinq chaînes de caractères au clavier, il les stocke dans les variables de la liste puis les restitue en ordre inverse.

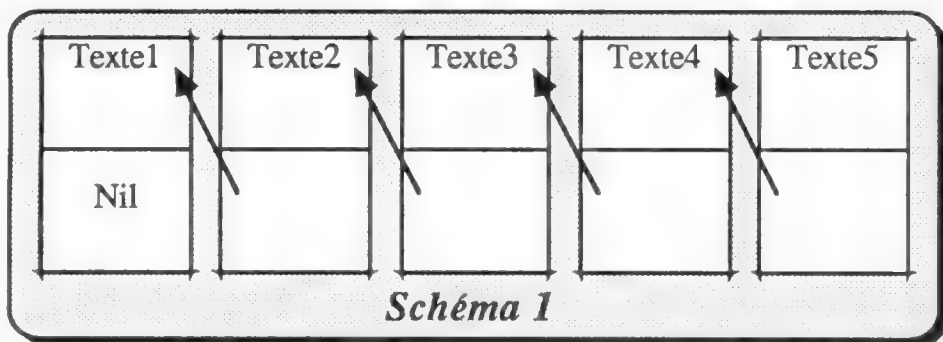
Analysons les différentes instructions.

Aucun commentaire spécial sur la partie déclarative du programme. Le type pointeur est bien défini avant le type des variables qu'il

### Programme 5

```
program liste;
    type
        ptr      = ^element
        element  = record
            info   : string[10];
            suivant : ptr
        end;
    var p, premier : ptr;
        i          : integer;
        texte      : string[10];
    begin
        writeln ('Creation de 5 elements');
        premier:=nil;
        for i:=1 to 5 do
            begin
                new(p);
                readln(texte);
                p^.info:=texte
                p^.suivant:=premier;
                premier:=p
            end;
        writeln ('Lecture des elements');
        p:=premier
        while p<>nil do
            begin
                writeln(p^.info);
                p:=p^.suivant
            end
        end
    end.
```





va désigner. P et PREMIER sont des pointeurs (statiques puisque déclarés) qui serviront à parcourir la liste. SUIVANT est un champ de ELEMENT (variable dynamique) qui référencera la variable précédente.

L'instruction **premier:=nil;** initialise le pointeur premier (il ne pointe sur rien). L'instruction **new(p);** crée une variable dynamique p<sup>^</sup> dont les deux champs sont définis par **p<sup>^</sup>.info:=texte** (chaîne lue au clavier) et **p<sup>^</sup>.suivant:=premier;**. La première variable dynamique (de type ELEMENT) contient donc un pointeur qui ne pointe sur rien. On affecte ensuite, par **premier:=p;**, à PREMIER la valeur de P qui, elle, pointe sur cette première variable que l'on vient de définir. Lors du second passage dans la boucle le champ 'suivant' de la seconde variable dynamique contiendra donc le pointeur référencant la première variable. De proche en proche on définit pour chacune des variables le champ 'suivant' comme référencant la variable qui la précède. La première variable n'ayant pas d'antécédent son pointeur vaut bien NIL.

Schématiquement on a :

Voir schéma 1

La suite du programme se contente de lire, dans le sens inverse de la définition, les différentes variables et affiche, pour chacune d'entre elles le champ 'info'.

**p:=premier;** affecte bien à p la valeur du dernier pointeur (cette valeur n'est contenue dans aucune variable: il s'agit de la

position de la cinquième variable). **writeln(p<sup>^</sup>.info)** affiche donc bien le champ info de la dernière variable. **p:=p<sup>^</sup>.suivant** affecte à p la position de la quatrième variable (qui est donc désormais p<sup>^</sup>), etc...

Il n'y a aucune limite à la complexité des structures que l'on peut ainsi construire. Il suffit de gérer plusieurs pointeurs dans les champs des éléments composant une liste.

Un exemple classique est la liste circulaire que l'on pourrait schématiser selon:

Voir schéma 2

## Destruction des variables dynamiques

Contrairement aux variables statiques, la durée de vie des variables dynamiques est indépendante de la procédure dans laquelle elles ont été créées. Pour détruire ces variables il convient d'effectuer une opération explicite.

L'instruction **dispose(P)** désalloue la variable P<sup>^</sup> : la valeur de

P devient alors indéfinie.

## Remarques

- Si la variable pointée est un article avec variante créée par **new(P,V1,V2)** il faut utiliser **dispose(P,V1,V2)** et avec les mêmes variantes que lors de sa création.

- La disparition de tout repère sur une variable pointée n'entraîne pas la désallocation de cette variable. Ainsi la séquence **new(P); P:=nil;** fait que la variable P<sup>^</sup> créée lors du new ne sera plus jamais accessible alors que l'espace qui lui a été alloué encombrera la mémoire jusqu'à la fin du programme.

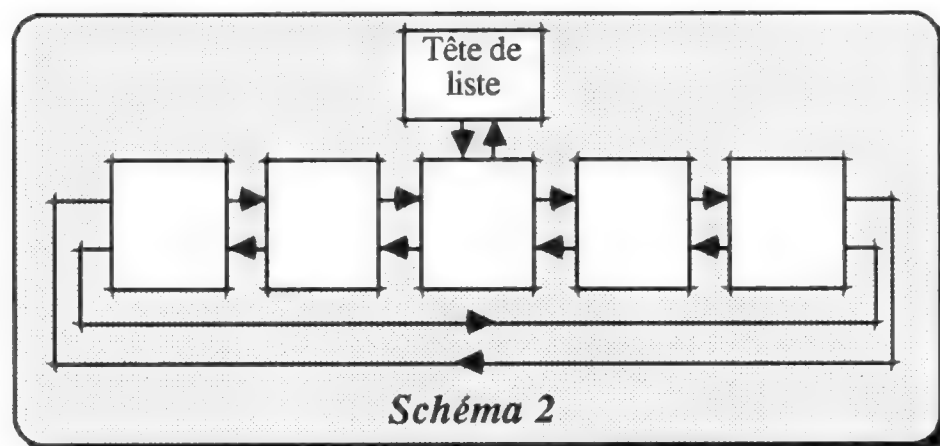
- On ne peut exécuter un **dispose(P)** si P est indéfini ou possède la valeur NIL.

## Exemple d'utilisation des pointeurs

Le petit programme suivant illustre l'une des utilisations des pointeurs pour la création et la mise à jour d'un ensemble de données dynamiques.

Voir programme 6

Ce programme lit un fichier (texte quelconque ou source PASCAL) et établit la liste des mots utilisés ainsi que leur nombre d'apparition. Comme on ne peut savoir en début d'exécution le nombre de mots différents contenus dans le texte, ce programme crée au fur et à mesure un dictionnaire dynamique répertoriant chaque mot.



## Programme 6

Suite du programme  
↓  
page suivante

```

program POINTEURS;
const
    lmax=15;    {taille maxi des identificateurs}

type
    ptelem = ^elem;
    elem = record
        suiv      : ptelem; {pointeur de la variable suivante}
        id         : string[15];
        n          : integer
    end;
    classe = (lettre,chiffre,quote,accolade,autre);

var
    fichier      : string[20];
    f             : file of char;
    categ         : array[char] of classe;
    ident         : string[15]; {identificateur courant}
    tete          : ptelem;     {tete de la liste}
    c             : char;       {caractere courant}

procedure fairecategorie;
var
    i             : 0..255;
    c             : char;

begin
    for i:=0 to 255 do categ[chr(i)]:=autre;
    for c:='A' to 'Z' do categ[c]:=lettre;
    for c:='a' to 'z' do categ[c]:=lettre;
    for c:='0' to '9' do categ[c]:=chiffre;
    categ['{']:=accolade;
    categ['}']:=accolade;
    categ['"']:=quote
end;

procedure impliste;
var
    p:ptelem;

begin
    page(output);
    writeln(":10,'Impression du dictionnaire');
    writeln(":10,'_____');
    writeln;
    p:=tete;
    while p<>nil do
        with p^ do
            begin
                writeln(":10,id,' ==> ',n:4);
                p:=suiv
            end
        end;
end;

```

Les variables dynamiques ainsi créées sont des articles sans variante (ELEM) ayant chacune trois champs:

- **suiv** qui est un pointeur référencant la variable suivante (il s'agit d'une liste où, comme nous l'avons vu, chaque variable pointe sur sa précédente)
- **id** qui est une chaîne de caractères représentant le mot à mémoriser
- **n** de type entier qui représente le nombre d'occurrences du mot dans tout le texte.

Pour pouvoir traiter les fichiers sources écrits en PASCAL ce programme ignore les identificateurs qui apparaissent dans des commentaires (c-a-d entre {}).

Le programme se compose

- d'une procédure d'initialisation (FAIRECATEGORIE) qui définit le type de chaque caractère rencontré. Notez au passage l'une des originalités de Pascal qui permet de créer des tableaux dont les identificateurs des éléments ne sont pas entiers mais de type ordinal de base **char**. CATEG est un tableau dont les éléments sont de type "classe" (tous les composants de ce type sont donnés explicitement) et dont l'indexation se fait par un caractère.
- d'une procédure d'impression de la liste (du dictionnaire). Cette procédure ne pose aucun problème particulier : elle est parcourue séquentiellement en fin de programme et affiche à l'écran tous les mots avec le nombre d'apparitions.

- d'une procédure d'identification du mot en cours de lecture (LIREIDENT). Cette procédure ignore donc les commentaires et poursuit la lecture du fichier d'entrée jusqu'à la fin du mot lorsqu'elle rencontre un caractère.

- d'une procédure de recherche en liste (CHERCHEID). Cette procédure parcourt séquentiellement la liste pour voir si le mot courant a déjà été référencé. Si tel est le cas, le champ *n* de la variable en question est incrémenté de 1, sinon une nouvelle variable dynamique est créée. Le champ *n* est de cette variable est mis à 1, le champ *suiv* pointe la variable suivante (il s'agit chronologiquement de la variable précédente). Le pointeur de tête de liste (celui qui définit le premier élément du dictionnaire) référence alors cette nouvelle variable.

En fait, chaque fois que l'on crée une variable dynamique elle prend la première position de la liste et pointe sur l'ancienne variable de tête. On peut schématiser ce processus par la figure suivante.

Voir schéma 3

Les structures dynamiques chaînées sont très souples et très puissantes. Néanmoins l'accès direct à un élément donné est impossible : il est nécessaire de parcourir séquentiellement la liste jusqu'à sa rencontre. De même l'insertion ou la suppression d'un élément au milieu d'une liste n'est pas très aisée puisqu'elle suppose la mise à jour des pointeurs.

Si le traitement que l'on doit faire sur un ensemble de données est global et ne nécessite pas l'accès direct ou sélectif à certains nombres de ses éléments l'emploi d'une liste est un excellent moyen pour sauvegarder de la mémoire.

Plus généralement l'emploi de variables dynamiques (pointées par des variables statiques ou dynamiques) est très intéressant dès qu'il s'agit de gérer de gros

## Programme 6 (suite)

```

procedure lireident;
var
    k      : 0..lmax;
    trouve : boolean;

begin
    trouve:=false;
    repeat
        case categ[c] of
            lettre : begin
                ident:= '      '; {15 blancs}
                k:=0;
                repeat
                    if k<lmax then
                        begin
                            k:=k+1;
                            ident[k]:=c
                        end;

                    read(f,c);
                    write(c);
                    until (categ[c]<>lettre) and (categ[c]<>chiffre);
                    trouve:=true;
                end;

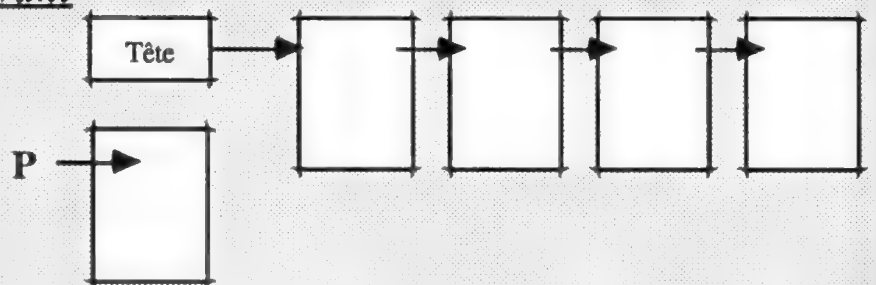
                accolade:repeat
                    read(f,c);
                    write(c);
                    until c='}';

                quote:repeat

```

Suite du programme  
page suivante ⇒

Avant



Après

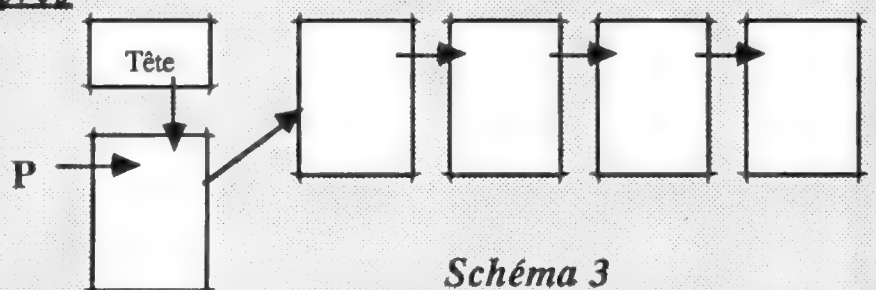


Schéma 3



## Programme 6 (suite et fin)

```

        read(f,c);
        write(c);
        until c="";

        chiffre,autre:begin
        read(f,c);
        write(c)
        end

        end {fin du case}

until trouve or eof(f)
end;

procedure cherchid;

var
        pastrouve:boolean;
        p:ptelem;

begin
        p:=tete;
        pastrouve:=true;
        while (p<>nil) and pastrouve do
                if p^.id=ident then begin
                        pastrouve:=false;
                        p^.n:=p^.n +1
                        end
                else p:=p^.suiv;
        end

        if pastrouve then
                begin
                        new(p);
                        p^.id:=ident;
                        p^.n:=1;
                        p^.suiv:=tete;
                        tete:=p;
                end
        end;
end;

begin {programme principal}
page(output);
fairecategorie;
tete:=nil;
{$I-}
        repeat
                write('FICHER ? ');
                readln(fichier);
                reset(f,fichier);
                until ioresult=0;
        {$I+}
        read(f,c);
        write(c);
        while not eof(f) do
                begin
                        lireident;
                        cherchid
                end;
        close(f);
        impliste
end.

```

volumes d'information dont on ne connaît pas la taille à priori. La plus importante restriction de cette technique vient du fait que sur le TI aucune vérification de la taille de la mémoire disponible n'est faite lors de la création d'une nouvelle variable. C'est donc au programmeur de s'assurer (au moyen de MEMAVAIL, VARAVAIL, VARDISPOSE et VARNEW) de l'intégrité de sa partition mémoire.

99

**99 MAGAZINE**

**vous attend à la  
boutique Sicob  
- Porte Maillot -  
au stand M279.**

# Musique et Mini-mémoire

Denise Amrouche

**S**i vous programmez souvent en Basic TI, vous vous êtes certainement heurtés au problème suivant : faire entendre de la musique pendant un jeu ou pendant l'installation d'un décor. Synchroniser les CALL SOUND et les autres instructions nécessite de nombreux tâtonnements.

## Source de la routine "Musique et Mini-mémoire"

\*RESERVER UNE PLACE EN VDP  
\*LONGUEUR DE LA LISTE DES SONS DANS >7118  
\*PREMIERE ADRESSE DE LA PLACE ALLOUEE DANS R0

```
7FBA      AORG >7FBA
7FBA C820  MOV @>7118,@>830C
7FBC 7118
7FBE 830C
7FC0 04E0  CLR @>837C
7FC2 837C
7FC4 0420  BLWP @>6018      (GPLLNK)
7FC6 6018
7FC8 0038  DATA >0038
7FCA C020  MOV @>831C,R0
7FCC 831C
```

\*CHARGER EN MEMOIRE VIDEO LA LISTE DES SONS

```
7FCE 0201  LI R1,>711A (1ERE ADRESSE DE LA LISTE)
7FD0 711A
7FD2 C0A0  MOV @>7118,R2 (LONGUEUR DE LA LISTE)
7FD4 7118
7FD6 0420  BLWP @>6028      (VMBW)
7FD8 6028
```

\*LANCER L'EXECUTION

```
7FDA 0300  LIM1 0
7FDC 0000
7FDE C800  MOV R0,@>83CC
7FE0 83CC
7FE2 0202  LI R2,>0100
7FE4 0100
7FE6 D802  MOVB R2,@>83CE
7FE8 83CE
7FEA F802  SOCB R2,@>83FD
7FEC 83FD
7FEE 0300  LIM1 2
7FF0 0002
7FF2 04E0  CLR @>837C
7FF4 837C
7FF6 045B  B *R11      (RETOUR)
7FF8 4D     TEXT 'MUSIQU'
7FFE 7FBA  DATA >7FBA
END
```

Ce programme Basic va vous permettre d'écrire en langage machine, la liste des sons à émettre, et d'implanter la routine pour exécuter ces sons dans la mini mémoire. Vous pourrez alors, à tout moment, directement ou dans un programme, en tapant **CALL LINK ("MUSIQU")**, faire exécuter votre morceau.

Toute la partie "DATA" du programme représente les notes à jouer. Elles sont fournies à peu près comme en Basic.

- durée multipliée ensuite par M (ici M = 167) à diminuer pour un rythme plus rapide ;
- fréquence, puissance du canal 1 ;
- fréquence, puissance du canal 2 ;
- fréquence, puissance du canal 3.

## Différences par rapport à CALL SOUND

Quelques différences par rapport au **CALL SOUND** du Basic : chaque ligne de "DATA" doit comporter 6 virgules ; sur un canal donné, une note commencée ne s'arrête que si une autre note est envoyée (éventuellement avec une puissance 30 si on ne veut pas l'entendre). On peut donc ne rien mettre comme fréquence et puissance si l'on veut réentendre la même note sur le même canal. La dernière "DATA" est par convention DATA 0 pour terminer le processus.

## Utilisation

Taper le programme Basic (les lignes de "DATA" peuvent être modifiées suivant l'air que vous voulez entendre).

Le sauvegarder, car d'une manière générale, lorsqu'un programme fait appel à une routine en assembleur, on risque l'erreur fatale.

Faire RUN et... patienter. On implante ainsi en fin de mini mémoire, la routine assembleur listée et commentée en fin d'article, puis la liste des sons codée comme il faut pour l'exécution.

Le programme indique s'il y a trop de "DATA" pour la capacité de la mini mémoire, ou le nombre d'octets occupés (3746 au maximum).

La liste des sons et la routine étant maintenant en mini mémoire, CALL LINK ("MUSIQU") provoquera à tout moment l'exécution (en occupant une partie de la mémoire vidéo). CALL SOUND ( -1, 400, 30), par exemple, provoquera si on le souhaite une interruption avant la fin.

99

## Programme de démonstration de la routine "Musique et Mini-mémoire"

```
100 REM ECRITURE DE
    MUSIQUE EN MINIM
    EM
110 REM *****PAR DENIS
    E AMROUCHE *****
120 REM CHARGEMENT
    EN MINI-MEMOIRE D
    E LA ROUTINE ASS
    EMBLEUR
130 CALL LOAD(32698,20
    0,32,113,24,131,12
    ,4,224)
```

```
140 CALL LOAD(32706,13
    1,124,4,32,96,24,0
    ,56)
150 CALL LOAD(32714,19
    2,32,131,28,2,1,11
    3,26)
160 CALL LOAD(32722,19
    2,160,113,24,4,32,
    96,40)
170 CALL LOAD(32730,3,
    0,0,0,200,0,131,20
    4)
180 CALL LOAD(32738,2,
    2,1,0,216,2,131,20
    6)
190 CALL LOAD(32746,24
    8,2,131,253,3,0,0,
    2)
200 CALL LOAD(32754,4,
    224,131,124,4,91)
210 REM MISE DU NOM
    DFDE LA ROUTINE
220 CALL LOAD(32760,77
    ,85,83,73,81,85,12
    7,182)
230 CALL LOAD(28700,12
    7,248,127,248)
240 REM -----
    -----
250 CALL CLEAR
260 REM LISTE DES SO
    NS:DUREE,FREQUEN
    CE1,PUISSANCE1,FRE
    QU2,PUIS2,FREQU3,P
    UIS3
270 REM LA DUREE EST
    MULTIPLIEE PAR M
280 M=167
290 REM UNE NOTE PAR
    LIGNE,6 VIRGULES P
    AR LIGNE
300 DATA 1,466,0,,,
310 DATA 1,466,0,,,
320 DATA 2,523,0,156,0
330 DATA 2,466,0,196,0
340 DATA 2,466,0,233,0
350 DATA 2,466,0,156,0
360 DATA 2,392,0,196,0
370 DATA 2,392,0,233,0
380 DATA 2,349,0,156,0
390 DATA 2,311,0,196,0
400 DATA 2,392,0,233,0
410 DATA 2,466,0,156,0
420 DATA 2,466,0,196,0
430 DATA 2,466,0,233,0
440 DATA 2,262,0,131,0
450 DATA 2,311,0,156,0
460 DATA 2,311,0,196,0
470 DATA 2,349,0,131,0
480 DATA 2,311,0,156,0
490 DATA 2,349,0,196,0
500 DATA 2,392,0,156,0
510 DATA 2,466,0,196,0
520 DATA 2,466,0,233,0
530 DATA 2,466,0,156,0
540 DATA 2,466,0,196,0
550 DATA 2,466,0,233,0
560 DATA 2,523,0,156,0
570 DATA 2,466,0,196,0
580 DATA 2,466,0,233,0
590 DATA 2,466,0,156,0
600 DATA 2,392,0,196,0
610 DATA 2,392,0,233,0
620 DATA 2,349,0,156,0
630 DATA 2,311,0,196,0
640 DATA 2,392,0,233,0
650 DATA 2,466,0,156,0
660 DATA 2,466,0,196,0
670 DATA 1,466,0,233,0
680 DATA 1,466,0,233,0
690 DATA 2,262,0,131,0
700 DATA 2,311,0,156,0
710 DATA 2,311,0,196,0
```



720 DATA 2,349,0,131,0	1030 DATA 2,466,0,156,0,,	1350 READ D
730 DATA 2,311,0,156,0	1040 DATA 2,466,0,196,0,,	1360 IF (D<0)+(D>4250/M) THEN 1710
740 DATA 2,349,0,196,0	1050 DATA 1,466,0,233,0,,	1370 IF D=0 THEN 1630
750 DATA 2,392,0,156,0	1060 DATA 1,466,0,233,0,,	1380 ADR=ADR+1
760 DATA 2,311,0,196,0	1070 DATA 2,523,0,156,0,,	1390 FOR V=1 TO 3
770 DATA 2,349,0,233,0	1080 DATA 2,466,0,196,0,,	1400 READ FR\$,PU\$
780 DATA 2,311,0,156,0	1090 DATA 2,466,0,233,0,,	1410 IF PU\$="" THEN 1510
790 DATA 2,311,0,196,0	1100 DATA 2,466,0,156,0,,	1420 F=VAL(FR\$)
800 DATA 2,311,0,233,0	1110 DATA 2,392,9,196,0,,	1430 P=VAL(PU\$)
810 DATA 2,262,0,131,0	1120 DATA 2,392,0,233,0,,	1440 IF (F<110)+(F>44733) THEN 1710
820 DATA 2,311,0,156,0	1130 DATA 2,349,0,156,0,,	1450 IF (P<0)+(P>30) THEN 1710
830 DATA 2,311,0,196,0	1140 DATA 2,311,0,196,0,,	1460 G=111860.8/F
840 DATA 2,262,0,131,0	1150 DATA 2,392,0,233,0,,	1470 G2=INT(G/16)
850 DATA 2,311,0,156,0	1160 DATA 2,466,0,156,0,,	1480 G1=G-16*G2
860 DATA 2,311,0,196,0	1170 DATA 2,466,0,196,0,,	1490 CALL LOAD(ADR,96+32*V+G1,G2,112+32*V+P/2)
870 DATA 2,294,0,147,0	1180 DATA 2,466,0,233,0,,	1500 ADR=ADR+3
880 DATA 2,311,0,156,0	1190 DATA 2,262,0,131,0,,	1510 NEXT V
890 DATA 2,311,0,156,0	1200 DATA 2,311,0,156,0,,	1520 D=D*M*6/100
900 DATA 2,466,0,196,0	1210 DATA 2,311,0,196,0,,	1530 IF D>1 THEN 1550
910 DATA 2,466,0,196,0	1220 DATA 2,349,0,131,0,,	1540 D=1
920 DATA 1,466,0,233,0	1230 DATA 2,311,0,156,0,,	1550 CALL LOAD(ADR,D)
930 DATA 1,466,0,233,0	1240 DATA 2,349,0,196,0,,	1560 ADR=ADR+1
940 DATA 2,262,0,131,0	1250 DATA 2,392,0,156,0,,	1570 DIF=ADR-ADR-2
950 DATA 2,311,0,156,0	1260 DATA 2,311,0,196,0,,	1580 CALL LOAD(ADR,DIF)
960 DATA 2,311,0,196,0	1270 DATA 2,349,0,233,0,,	1590 PRINT I
970 DATA 2,349,0,131,0	1280 DATA 4,311,0,196,0,,	1600 I=I+1
980 DATA 2,311,0,156,0	1290 DATA 2,311,0,196,0,,	1610 ADR=ADR
990 DATA 2,349,0,196,0	1300 DATA 2,311,0,196,30,,	1620 IF ADR>32780 THEN
1000 DATA 2,392,0,156,0,,	1310 DATA 0	1730 ELSE 1350
1010 DATA 2,466,0,196,0,,	1320 REM LA LISTE SE	1630 CALL LOAD(ADR,4,159,191,223,255,0)
1020 DATA 2,466,0,233,0,,	TERMINE TOUJOURS PAR DATA 0	1640 BY=ADR+6-DE
		1650 BY1=INT(BY/256)
		1660 CALL LOAD(28952,BY1,BY-256*BY1)
		1670 PRINT "NOMBRE DE BYTES UTILISES :";BY+2
		1680 CALL LINK("MUSIQUE")
		1690 STOP
		1700 REM MESSAGE D'ERREUR DANS LES DATA
		1710 PRINT "ERREUR DANS LA DATA NO";I+1
		1720 STOP
		1730 PRINT "JE NE PEUX PLUS CHARGER AU DELA DE LA DATA NO :";I
		1740 PRINT "CAR JE N'AI PLUS DE PLACE DANS LA MINI-MEMOIRE"
	1330 DE=28954	
	1340 AD=28954	

# Transfert Image/Cassette

Gérard Baroni et Jean Marin

**A**près vous avoir indiqué brièvement comment, avec la norme Texas Instruments, un transfert d'image vers une cassette - ou inversement - peut s'effectuer, nous exposerons notre solution qui apporte un gain substantiel de temps, de place mémoire et qui, de plus, donne accès à tous les modes graphiques du TI-99/4A.

La configuration exigée comprend la Mini-mémoire ou le Basic étendu plus l'extension 32Ko.

Voyons d'abord la manière de procéder en utilisant la norme Texas.

- En mode standard, il s'agit de réaliser une copie d'écran et de sauvegarder les codes ASCII des 768 cases de l'écran. Pour le transfert de ces 768 octets vers la cassette à l'aide de OPEN # CS1 et PRINT #..., 192 octets sont transférés toutes les 10 secondes. 40 secondes sont nécessaires pour expédier le contenu de l'écran vers notre cassette.
- Pour sauvegarder les tables d'écrans, des couleurs et des formes, soit 2048 octets au maximum, 11 enregistrements sont nécessaires et le transfert dure environ 2 minutes.
- En BitMap, où nous avons 14 Ko d'image et 2 Ko de Basic restant, il y a interférence entre les "buffers" cassette et le reste. Avec 75 enregistrements de 10 secondes, il faudrait environ 12 minutes pour transférer l'image.

## Notre solution

Pour disposer d'une plus grande liberté, nous allons nous brancher sur les routines ROM de la console sans passer par les routines GPL qui sont dans les GROMS. Les avantages sont les suivants :

- 1) Le temps de silence pour l'amorce (avant le sifflement) disparaît.
- 2) Le transfert direct de l'image vers la cassette - et inversement - se fait sans passer par un buffer. Donc, il n'y a pas de place perdue en mémoire. A la lecture, l'image apparaît au fur et à mesure du déroulement de la cassette.

On peut aller *en un seul enregistrement* jusqu'à un maximum de 255 fois 64 octets soit 16 320 octets, c'est à dire la totalité de la mémoire vive vidéo moins 64 octets.

## Temps de transfert

- Table écran (768 octets) :  
15 secondes au lieu de 40 ;
- Image (2048 octets) :  
30 secondes au lieu de 120 ;
- Image BitMap (14Ko) :  
3'15" au lieu de 12'30".

Les programmes "SI" (Sauve Image) et "LI" (Lire Image), en assembleur dont les listings suivent, sont accessibles du Basic par des CALL LINK("SI") ou ("LI").

Après exécution, le retour au Basic se fait automatiquement, étant prévu dans la routine console.

## Important

Dans le mode graphique 2 Ko, le CALL LINK ("LI") doit être précédé de CALL CHAR (159,"") qui a pour effet de réserver 2 Ko d'image.

En BitMap, le CALL LINK ("LI") doit être précédé d'un CALL LOAD (-31890,56,0) qui réserve 14 Ko d'image et d'un CALL LINK ("BITMAP") - voir 99 magazine précédent - qui initialise le mode graphique BitMap dans le processeur vidéo.

## Méthode de transfert employée par Texas Instruments

Les programmes "LI" et "SI" réalisent des transferts d'informations, mais étudions un peu le mécanisme de transfert prévu par le constructeur.

Nous savons que tout message est constitué par une suite d'informations élémentaires codées sous forme binaire (0, 1). Ainsi, lors de l'enregistrement sur cassette ("SAVE CS1"), chacune de ces informations sera représentée par un signal d'une durée déterminée permettant à la machine de distinguer, au moment de la lecture ("OLD CS1"), entre la valeur 0 ou 1 du bit enregistré. Dans le système Texas, les mesures effectuées à l'aide d'un oscilloscope nous donnent les résultats suivants :

- lorsqu'il s'agit du bit de valeur 0, la durée (t) du signal est approximativement de 0,74 milliseconde (ms) ;
- dans le cas du bit de valeur 1, (t) correspond à 2 phases d'émission séparées par un pic d'environ 0,37 ms chacune (t/2). La fréquence du signal enregistré est de l'ordre de 1400 bits/sec ou 10 Ko/minute.

Examinons maintenant le contenu de l'enregistrement d'un programme sur cassette. Au début de celui-ci, nous entendons un sifflement pendant plus de 14", ce qui équivaut au passage de 768 octets de valeur 0. Puis, survient 1 octet de valeur >FF annonçant le début d'un message, puis 2 octets identiques de valeur N (N désigne le nombre de tranches de 64 octets qui vont

constituer le message proprement dit ; ce nombre varie en fonction de la longueur du programme).

Tout ceci précède N groupes de 74 octets répétés 2 fois. Dans chacun de ces groupes, les octets se répartissent ainsi :

- 8 octets à 0 ;
- 1 octet à >FF ;
- 1 paquet de 64 octets correspondant au message ;
- 1 octet "checksum", ou somme modulo 256, des 64 octets permettant à la machine de tester la présence ou non d'erreurs dans un groupe.

Les 8 octets de valeurs 0 et l'octet de valeur >FF sont très importants : ils mesurent la vitesse du magnétophone et permettent une adaptation systématique asservissement de vitesse, et

démontrent la haute fiabilité du système.

Le "checksum" est un octet dont la valeur est calculée à l'enregistrement. Lors de la lecture, cette valeur sera comparée et devra être égale au reste de la division entière par 256 de la somme des valeurs des 64 octets. Comme nous savons que la valeur d'un octet ne saurait dépasser 255, si la somme des valeurs des 64 octets est 260 par exemple, le "checksum" devra être égal à 4 (260 modulo 256 = 4).

La répétition des groupes de 74 octets permet donc de corriger une éventuelle erreur. En effet, lorsqu'un bit est faux, le "checksum" ne correspondra pas à la somme modulo 256 et, par conséquent, l'ordinateur saura que cette tranche est erronée. C'est pourquoi il prendra l'autre tranche en totalité, à condition qu'elle soit bonne. Dans le cas contraire, il affichera "DATA ERROR". Par analogie, on peut comparer le TI à un véhicule disposant d'une roue de secours, qui sera utilisée lors d'une crevaison.

Nous voyons combien ce dispositif ralentit la transmission des données puisqu'il y a en fait 148 octets pour un message réel de 64 octets seulement. La vitesse moyenne est de 600 bits/seconde, soit 75 octets/seconde, soit 4,5 Ko/minute.

Les gens pressés peuvent modifier les routines en ROM, après translation dans la RAM de la Mini-mémoire, pour supprimer la duplication des tranches, réalisant ainsi un gain de temps appréciable (t/2). Ils peuvent même accélérer le signal dans un rapport 2 si le magnétophone le permet.

On arrive ainsi à transférer une image BitMap de 14 Ko en 35 secondes environ. La même technique permet aussi de charger l'extension mémoire rapidement.

## Source des routines SI" et "LI"

### "SI"

		AORG	>XXXX	* Adresse départ
				* à choisir
XXXX	02E0	LWPI	>83E0	
	83E0			
	0201	LI	R1, TA	
	XXXX			
	0460	B	@>1346	* Vers routine
	1346			
XXXX	XXXX	TA	DATA	N
				* Nombre d'octets
				* à transférer
	XXXX		DATA	A
				* Adresse de base
				* en RAM vidéo
				* (0 : normal)

### "LI"

XXXX	02E0	LWPI	>83E0	
	83E0			
	0201	LI	R1, TA	
	XXXX			
	0460	B	@>142E	
XXXX	XXXX	TA	DATA	N
	XXXX		DATA	A

NE PAS OUBLIER LA TABLE REF/DEF



Votre assiduité méritant récompense, voici joint le listing d'un programme éditeur de dessin en BitMap qui vous permettra de créer et de sauvegarder sur cassette vos graphismes. Ce programme utilise les routines vues dans les numéros précédents de 99 magazine.

NB : En cas de modification du programme, souvenez-vous que vous ne disposez que de 2 Ko pour le Basic et que les REMs et les noms de variables trop longs sont à proscrire.

**99**

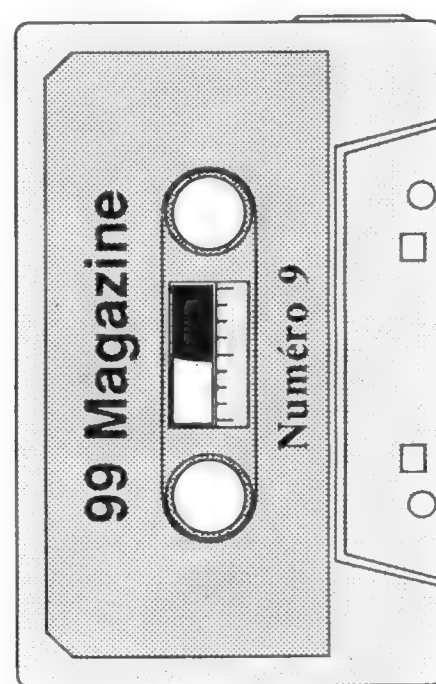
## Routines "SI" et "LI" Programme de démonstration

### Basic TI et Mini-mémoire

```
100 CALL LOAD(-31890,0
)
110 CALL CLEAR
120 PRINT " LES COMMAN
DES SONT:": "-D,S,
E,X POUR DEPLACER"
: "          LE CURS
EUR": :
130 PRINT "-R          DE
PLACEMENT RAPIDE":
: "-L          DEPLACE
MENT LENT": :
140 PRINT "-C          CO
ULEUR": "-BEGIN
DEPART DU SEGMENT"
: "          A TRACE
R": "-ENTER TRAC
E DU SEGMENT": :
150 PRINT "-AID        SA
UVER SUR K7": "-BA
CK      RETOUR AU BA
SIC": :
```

```
160 INPUT "(ENTER) PO
UR COMMENCER": R$
170 CALL CLEAR
180 PRINT "VOULEZ-VOUS
RETROUVER UN": "D
ESSIN SUR K7 ";
190 INPUT "(O/N)? ": R$
200 CALL LINK("BITMAP"
)
210 IF R$<>"O" THEN 23
0
220 CALL LINK("LI")
230 D=1
240 C=240
250 Y=96
260 X=128
270 CALL POKEV(6912,Y-
3,X-3,128,C/16,208
,"",7168,0,68,56,4
0,56,68,0,0)
280 YD=Y
290 XD=X
300 CALL POKEV(6912,Y-
3,X-3,128,C/16)
310 CALL KEY(5,R,E)
320 IF E=0 THEN 310
330 IF R<>69 THEN 370
340 Y=Y-D
350 Y=Y-D*(Y<0)
360 GOTO 300
370 IF R<>88 THEN 410
380 Y=Y+D
390 Y=Y+D*(Y>191)
400 GOTO 300
410 IF R<>83 THEN 450
420 X=X-D
430 X=X-D*(X<8)
440 GOTO 300
450 IF R<>68 THEN 490
460 X=X+D
470 X=X+D*(X>255)
480 GOTO 300
490 IF R<>67 THEN 520
500 C=C+16+240*(C=240)
510 GOTO 300
520 IF R<>14 THEN 560
```

```
530 XD=X
550 YD=Y
560 IF R<>13 THEN 610
570 CALL LINK("COLOR",
C)
580 CALL LINK("DROITE"
,YD,XD,Y,X)
590 CALL SOUND(50,440,
0)
600 GOTO 280
610 IF R<>82 THEN 650
620 CALL SOUND(50,1000
,0)
630 D=8
640 GOTO 310
650 IF R<>76 THEN 690
660 CALL SOUND(50,1000
,0)
670 D=1
680 GOTO 310
690 IF R<>1 THEN 720
700 CALL LINK("SI")
710 GOTO 310
720 IF R<>15 THEN 310
730 CALL PEEKV(-32768,
X)
```



**N**ous vous proposons ici un jeu en trois dimensions, pour deux joueurs. Vous pouvez évidemment jouer seul, mais cela n'a pas grand intérêt, sinon celui de se familiariser avec le maniement du clavier et s'entraîner.

## Règles du jeu

Chaque joueur dispose de douze étoiles réparties sur quatre plateaux. Pour gagner une partie, il faut être le premier à réaliser un alignement de quatre étoiles. Mais attention, pas n'importe quel alignement... vous devez retrouver celui choisi par l'ordinateur parmi les 68 possibilités offertes.

Les déplacements en diagonale sont interdits, vous devez toujours suivre lignes ou colonnes. Le programme refusera votre jeu si vous cherchez à tricher.

Vous pouvez également changer de plateau pour monter ou descendre vos étoiles, à condition que cette opération respecte la règle suivante : la nouvelle

position doit être située immédiatement à la verticale de l'ancienne, et libre de toute étoile.

Il est possible de gêner considérablement le jeu adverse puisque les permutations d'étoiles entre joueurs sont réalisables. Ce système permet d'employer plusieurs tactiques de jeu. A vous de les découvrir pour les utiliser au bon moment, et remporter la partie. Les permutations sont impossibles quand on change de plateau (le jeu serait trop facile).

Chaque étoile bien placée donne un "bip" sonore aigü.

## Jeu à partir du clavier

### Touches 1 à 4 (plateau)

Numéro de plateau. Ces numéros changent de couleur selon le tour du joueur (fond rouge ou blanc).

### Touches 1 à 4 (étoile)

Coordonnées de l'étoile. Numéro de ligne en premier (sur la gauche

# Starline

Georges Goument

du plateau) puis numéro de colonne (chiffres devant le plateau). L'étoile change de couleur.

### Touches 1 à 4 (coordonnées)

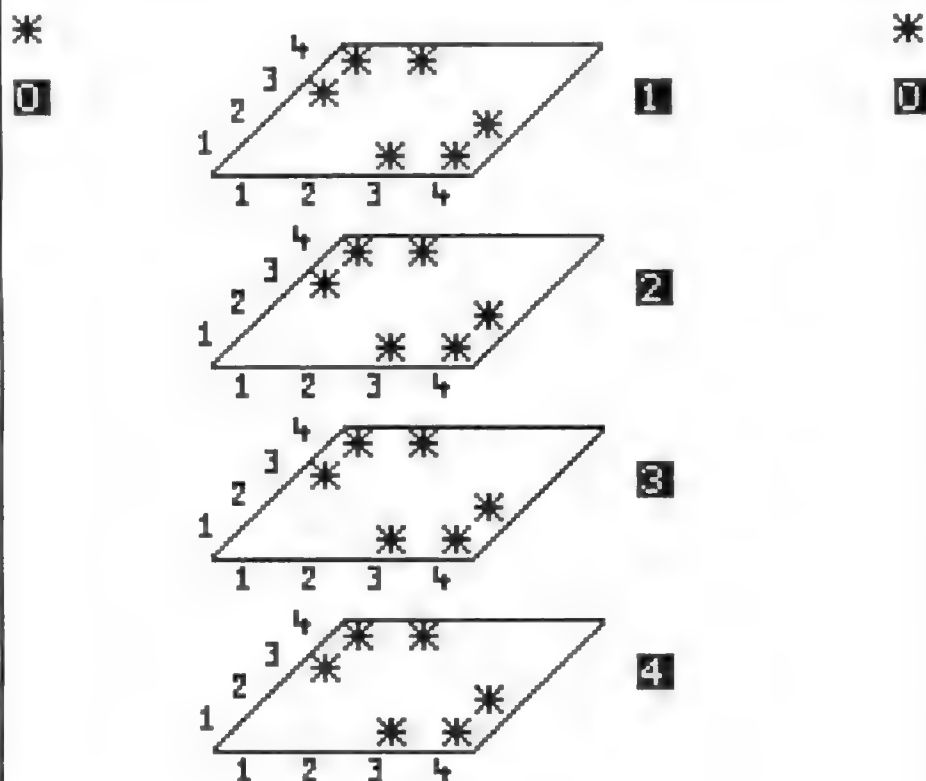
Tapez les nouvelles coordonnées : 1 à 4. Si vous ne changez pas de plateau de jeu, l'étoile occupe sa nouvelle position ou permute avec une étoile adverse.

### Touche N

Changement de plateau : touche N, puis 1 à 4 = nouveau numéro de plateau ; plus 1 à 4 pour la nouvelle position.

La touche N sert également pour l'annulation d'un coup, afin de jouer une autre position ou un autre plateau.

99



## Programme Starline

### Basic TI

```

10 REM *****
11 REM *
12 REM * STARLINE *
13 REM *
14 REM * Basic TI *
15 REM *
16 REM * copyright *
17 REM *
18 REM * 99 Magazine *
19 REM *
20 REM * G. Goument *
21 REM *****
22 G1=48
23 G2=48
24 CALL CLEAR
25 RANDOMIZE
26 DIM A(4)
27 J=0
28 AL=0
29 FOR I=33 TO 47
30 READ A$

```

```

31 CALL CHAR(I,A$)
32 NEXT I
33 DATA 000000000000206
02,020207,000000000
0070501,020407,0000
000000070103,010204
081020408
34 DATA 000000000000000
FF,FF00010301010103
,FF00000000000008,FF
00030200010203
35 DATA FF0080808000000
8,FF00030001000003,
FF0080808080808,FF0
002020203,FF0000008
0C0808
36 CALL SCREEN(14)
37 CALL COLOR(1,12,1)
38 CALL COLOR(2,12,1)
39 FOR I=9 TO 12
40 CALL COLOR(I,15,1)
41 NEXT I
42 CALL COLOR(3,15,1)
43 CALL CHAR(64,"92543
8FE385492")
44 CALL CHAR(72,"92543
8FE385492")
45 CALL CHAR(88,"92543
8FE385492")
46 CALL CHAR(80,"01010
7")
47 CALL CHAR(81,"00000
0000008080A")
48 CALL CHAR(82,"0F020
2")
49 INPUT "1 ou 2 joueu
rs ":N$
50 CALL CLEAR
51 FOR I=1 TO 8
52 CALL COLOR(I,1,1)
53 NEXT I
54 PRINT TAB(9);"Q ' '
' ' ' '":TAB(8);"R&@
@
&":TAB(7);"#P
&@
& 1":TAB(6
);"!$&
&"
55 PRINT TAB(6);" &
&":TAB(7);"()*+
,-./":TAB(9);"Q ' '
' ' ' '":TAB(8);"R&
&":TAB(7);"#P
&
& 2"
56 PRINT TAB(6);"!$&
@&":TAB(6);" &
@ @&":TAB(7);"()
*+,-./":TAB(9);"Q '
' ' ' ' '":TAB(8);"R
&@ @
&"
57 PRINT TAB(7);"#P&@
& 3":TAB(6);"
!$&
&":TAB(6)
;" &
&":TAB(7
);"()*+,-./":TAB(9)
;"Q ' ' ' ' '
"
58 PRINT TAB(8);"R&
&":TAB(7);"#P&
& 4":TAB(6)
;"!$&
@&":TAB(
6);" &
@ @&"
59 C=40
60 FOR I=9 TO 16
61 CALL HCHAR(24,I,C)
62 C=C+1
63 NEXT I
64 FOR I=5 TO 23 STEP
6
65 CALL HCHAR(I,8,34)
66 NEXT I
67 CALL HCHAR(1,3,72)
68 CALL HCHAR(1,30,64)
69 CALL HCHAR(3,3,G1)
70 CALL HCHAR(3,30,G2)
71 CALL COLOR(8,8,1)
72 CALL COLOR(5,16,1)
73 CALL COLOR(6,10,1)
74 CALL COLOR(7,12,1)
75 CALL COLOR(1,12,1)
76 CALL COLOR(2,12,1)
77 IF N$="1" THEN 84
78 DATA 4,17,5,14,5,16
,8,13,8,15,9,12,16,
17,17,14,17,16,20,1
3,20,15,21,12,0,0
79 RESTORE 78
80 READ Y,X
81 IF Y=0 THEN 84
82 CALL HCHAR(Y,X,72)
83 GOTO 80
84 AL=0
85 RA=0
86 RA=INT((68-1+1)*RND
)+1
87 N=0
88 Y=0
89 X=0
90 NO=3000
91 IF N$="1" THEN 93
92 IF J=1 THEN 97
93 J=1
94 V=16
95 H=64
96 GOTO 100
97 V=10
98 J=0
99 H=72
100 CALL COLOR(3,2,V)
101 CALL COLOR(4,2,V)
102 CALL KEY(3,T,S)
103 IF N=0 THEN 106
104 IF N>0 THEN 139
105 GOTO 127
106 IF S=0 THEN 102
107 IF T<>49 THEN 111
108 N=1
109 Z=3
110 GOTO 122
111 IF T<>50 THEN 115
112 N=2
113 Z=9
114 GOTO 122
115 IF T<>51 THEN 119
116 N=3
117 Z=15
118 GOTO 122
119 IF T<>52 THEN 124
120 N=4
121 Z=21
122 CALL SOUND(5,123,5
)
123 GOTO 102
124 IF T<>78 THEN 102
125 IF XA=0 THEN 127
126 CALL HCHAR(YA,XA,H
)
127 FOR I=1 TO 7
128 CALL SOUND(10,110,
0)
129 NEXT I
130 Y=0
131 X=0
132 N=0
133 Z=0
134 YA=0
135 XA=0
136 NA=0
137 O=0
138 GOTO 102
139 CALL KEY(3,T,S)
140 IF NA=0 THEN 145
141 IF N=NA+1 THEN 145
142 IF N=NA-1 THEN 145
143 IF N=NA THEN 145
144 GOTO 125
145 CALL HCHAR(Z,22,32
)
146 IF N<>1 THEN 149
147 CALL HCHAR(Z,22,49
)
148 GOTO 156
149 IF N<>2 THEN 152
150 CALL HCHAR(Z,22,50
)
151 GOTO 156
152 IF N<>3 THEN 155
153 CALL HCHAR(Z,22,51
)
154 GOTO 156

```



155 CALL HCHAR(Z,22,52 )	210 IF T<>52 THEN 198	266 CALL GCHAR(Y,X,E)
156 IF S=0 THEN 139	211 X=17	267 IF J=0 THEN 270
157 IF T<>78 THEN 161	212 GOTO 264	268 IF E=72 THEN 127
158 N=0	213 IF T<>51 THEN 239	269 GOTO 271
159 CALL SOUND(5,123,10)	214 IF N=1 THEN 223	270 IF E=64 THEN 127
160 GOTO 102	215 IF N=2 THEN 219	271 IF E=32 THEN 127
161 IF T<>49 THEN 187	216 IF N=3 THEN 221	272 CALL HCHAR(Y,X,88)
162 IF N=1 THEN 171	217 Y=21	273 YA=Y
163 IF N=2 THEN 167	218 GOTO 224	274 XA=X
164 IF N=3 THEN 169	219 Y=9	275 NA=N
165 Y=23	220 GOTO 224	276 O=1
166 GOTO 172	221 Y=15	277 Y=0
167 Y=11	222 GOTO 224	278 X=0
168 GOTO 172	223 Y=3	279 GOTO 102
169 Y=17	224 CALL SOUND(5,123,5 )	280 IF N<>NA THEN 291
170 GOTO 172	225 CALL KEY(3,T,S)	281 IF Y<>YA THEN 285
171 Y=5	226 IF S=0 THEN 225	282 IF X=XA+2 THEN 298
172 CALL SOUND(5,123,5 )	227 IF T<>49 THEN 230	283 IF X=XA-2 THEN 298
173 CALL KEY(3,T,S)	228 X=12	284 GOTO 125
174 IF S=0 THEN 173	229 GOTO 264	285 IF Y<>YA-1 THEN 288
175 IF T<>49 THEN 178	230 IF T<>50 THEN 233	286 IF X=XA+1 THEN 298
176 X=10	231 X=14	287 GOTO 125
177 GOTO 264	232 GOTO 264	288 IF Y<>YA+1 THEN 125
178 IF T<>50 THEN 181	233 IF T<>51 THEN 236	289 IF X=XA-1 THEN 298
179 X=12	234 X=16	290 GOTO 125
180 GOTO 264	235 GOTO 264	291 IF X<>XA THEN 125
181 IF T<>51 THEN 184	236 IF T<>52 THEN 224	292 IF Y=YA-6 THEN 295
182 X=14	237 X=18	293 IF Y=YA+6 THEN 295
183 GOTO 264	238 GOTO 264	294 GOTO 125
184 IF T<>52 THEN 172	239 IF T<>52 THEN 139	295 CALL GCHAR(Y,X,F)
185 X=16	240 IF N=1 THEN 249	296 IF F<>32 THEN 125
186 GOTO 264	241 IF N=2 THEN 247	297 GOTO 307
187 IF T<>50 THEN 213	242 IF N=3 THEN 245	298 CALL GCHAR(Y,X,F)
188 IF N=1 THEN 197	243 Y=20	299 IF F=32 THEN 307
189 IF N=2 THEN 193	244 GOTO 250	300 IF J=0 THEN 304
190 IF N=3 THEN 195	245 Y=14	301 CALL HCHAR(Y,X,64)
191 Y=22	246 GOTO 250	302 CALL HCHAR(YA,XA,72)
192 GOTO 198	247 Y=8	303 GOTO 313
193 Y=10	248 GOTO 250	304 CALL HCHAR(Y,X,72)
194 GOTO 198	249 Y=2	305 CALL HCHAR(YA,XA,64)
195 Y=16	250 CALL SOUND(5,123,5 )	306 GOTO 313
196 GOTO 198	251 CALL KEY(3,T,S)	307 IF J=0 THEN 311
197 Y=4	252 IF S=0 THEN 251	308 CALL HCHAR(Y,X,64)
198 CALL SOUND(5,123,5 )	253 IF T<>49 THEN 256	309 CALL HCHAR(YA,XA,32)
199 CALL KEY(3,T,S)	254 X=13	310 GOTO 313
200 IF S=0 THEN 199	255 GOTO 264	311 CALL HCHAR(Y,X,72)
201 IF T<>49 THEN 204	256 IF T<>50 THEN 259	312 CALL HCHAR(YA,XA,32)
202 X=11	257 X=15	313 CALL SOUND(10,200,10)
203 GOTO 264	258 GOTO 264	314 CALL SOUND(10,220,10)
204 IF T<>50 THEN 207	259 IF T<>51 THEN 262	315 CALL SOUND(10,240,10)
205 X=13	260 X=17	
206 GOTO 264	261 GOTO 264	
207 IF T<>51 THEN 210	262 IF T<>52 THEN 250	
208 X=15	263 X=19	
209 GOTO 264	264 CALL SOUND(5,123,5 )	
	265 IF O=1 THEN 280	

316 CALL SOUND (10,260,10)	374 IF RA=11 THEN 390	433 E=2
317 O=0	375 IF RA=12 THEN 394	434 F=15
318 Z=0	376 IF RA=13 THEN 398	435 GOSUB 441
319 XA=0	377 IF RA=14 THEN 402	436 GOTO 433
320 YA=0	378 IF RA=15 THEN 406	437 E=2
321 NA=0	379 IF RA=16 THEN 410	438 F=17
322 IF RA>64 THEN 531	380 IF RA=17 THEN 414	439 GOSUB 441
323 IF RA>60 THEN 452	381 IF RA=18 THEN 418	440 GOTO 437
324 IF RA>56 THEN 422	382 E=2	441 FOR I=1 TO 4
325 IF RA>40 THEN 616	383 F=17	442 IF AL=0 THEN 446
326 IF RA>=30 THEN 561	384 GOSUB 705	443 CALL HCHAR(E,F,88)
327 IF RA>=20 THEN 482	385 GOTO 382	444 GOSUB 738
328 IF RA>=10 THEN 373	386 E=15	445 GOTO 447
329 IF RA=1 THEN 369	387 F=12	446 CALL GCHAR(E,F,A(I))
330 IF RA=2 THEN 341	388 GOSUB 695	447 E=E+7
331 IF RA=3 THEN 345	389 GOTO 386	448 F=F-1
332 IF RA=4 THEN 349	390 E=16	449 NEXT I
333 IF RA=5 THEN 353	391 F=11	450 IF AL=1 THEN 777
334 IF RA=6 THEN 357	392 GOSUB 695	451 GOTO 752
335 IF RA=7 THEN 361	393 GOTO 390	452 IF RA=61 THEN 459
336 IF RA=8 THEN 365	394 E=17	453 IF RA=62 THEN 463
337 E=14	395 F=10	454 IF RA=63 THEN 467
338 F=13	396 GOSUB 695	455 E=5
339 GOSUB 695	397 GOTO 394	456 F=10
340 GOTO 337	398 E=20	457 GOSUB 471
341 E=3	399 F=13	458 GOTO 455
342 F=12	400 GOSUB 695	459 E=5
343 GOSUB 695	401 GOTO 398	460 F=16
344 GOTO 341	402 E=21	461 GOSUB 471
345 E=4	403 F=12	462 GOTO 459
346 F=11	404 GOSUB 695	463 E=5
347 GOSUB 695	405 GOTO 402	464 F=14
348 GOTO 345	406 E=22	465 GOSUB 471
349 E=5	407 F=11	466 GOTO 463
350 F=10	408 GOSUB 695	467 E=5
351 GOSUB 695	409 GOTO 406	468 F=12
352 GOTO 349	410 E=23	469 GOSUB 471
353 E=8	411 F=10	470 GOTO 467
354 F=13	412 GOSUB 695	471 FOR I=1 TO 4
355 GOSUB 695	413 GOTO 410	472 IF AL=0 THEN 476
356 GOTO 353	414 E=2	473 CALL HCHAR(E,F,88)
357 E=9	415 F=13	474 GOSUB 738
358 F=12	416 GOSUB 705	475 GOTO 477
359 GOSUB 695	417 GOTO 414	476 CALL GCHAR(E,F,A(I))
360 GOTO 357	418 E=2	477 E=E+5
361 E=10	419 F=15	478 F=F+1
362 F=11	420 GOSUB 705	479 NEXT I
363 GOSUB 695	421 GOTO 418	480 IF AL=1 THEN 777
364 GOTO 361	422 IF RA=57 THEN 429	481 GOTO 752
365 E=11	423 IF RA=58 THEN 433	482 IF RA=27 THEN 523
366 F=10	424 IF RA=59 THEN 437	483 IF RA=26 THEN 519
367 GOSUB 695	425 E=2	484 IF RA=25 THEN 515
368 GOTO 365	426 F=19	485 IF RA=24 THEN 511
369 E=2	427 GOSUB 441	486 IF RA=23 THEN 507
370 F=13	428 GOTO 425	487 IF RA=22 THEN 503
371 GOSUB 695	429 E=2	488 IF RA=21 THEN 499
372 GOTO 369	430 F=13	489 IF RA=20 THEN 495
373 IF RA=10 THEN 386	431 GOSUB 441	
	432 GOTO 429	

490 IF RA=28 THEN 527	549 GOTO 546	607 GOTO 604
491 E=20	550 FOR I=1 TO 4	608 E=14
492 F=13	551 IF AL=0 THEN 555	609 F=19
493 GOSUB 705	552 CALL HCHAR(E,F,88)	610 GOSUB 716
494 GOTO 491	553 GOSUB 738	611 GOTO 608
495 E=2	554 GOTO 556	612 E=20
496 F=19	555 CALL GCHAR(E,F,A(I	613 F=19
497 GOSUB 705	))	614 GOSUB 716
498 GOTO 495	556 E=E+6	615 GOTO 612
499 E=8	557 F=F+2	616 IF RA=41 THEN 635
500 F=13	558 NEXT I	617 IF RA=42 THEN 639
501 GOSUB 705	559 IF AL=1 THEN 777	618 IF RA=43 THEN 643
502 GOTO 499	560 GOTO 752	619 IF RA=44 THEN 647
503 E=8	561 IF RA=38 THEN 608	620 IF RA=45 THEN 651
504 F=15	562 IF RA=38 THEN 608	621 IF RA=46 THEN 655
505 GOSUB 705	563 IF RA=37 THEN 604	622 IF RA=47 THEN 659
506 GOTO 503	564 IF RA=36 THEN 600	623 IF RA=48 THEN 663
507 E=8	565 IF RA=35 THEN 596	624 IF RA=49 THEN 667
508 F=17	566 IF RA=34 THEN 592	625 IF RA=50 THEN 671
509 GOSUB 705	567 IF RA=33 THEN 588	626 IF RA=51 THEN 675
510 GOTO 507	568 IF RA=32 THEN 584	627 IF RA=52 THEN 679
511 E=8	569 IF RA=31 THEN 580	628 IF RA=53 THEN 683
512 F=19	570 IF RA=30 THEN 576	629 IF RA=54 THEN 687
513 GOSUB 705	571 IF RA=39 THEN 612	630 IF RA=55 THEN 691
514 GOTO 511	572 E=20	631 E=5
515 E=14	573 F=13	632 F=19
516 F=13	574 GOSUB 727	633 GOSUB 743
517 GOSUB 705	575 GOTO 572	634 GOTO 631
518 GOTO 515	576 E=20	635 E=2
519 E=14	577 F=15	636 F=13
520 F=15	578 GOSUB 705	637 GOSUB 743
521 GOSUB 705	579 GOTO 576	638 GOTO 635
522 GOTO 519	580 E=20	639 E=2
523 E=14	581 F=17	640 F=15
524 F=17	582 GOSUB 705	641 GOSUB 743
525 GOSUB 705	583 GOTO 580	642 GOTO 639
526 GOTO 523	584 E=20	643 E=2
527 E=14	585 F=19	644 F=17
528 F=19	586 GOSUB 705	645 GOSUB 743
529 GOSUB 705	587 GOTO 584	646 GOTO 643
530 GOTO 527	588 E=2	647 E=2
531 IF RA=65 THEN 538	589 F=13	648 F=19
532 IF RA=66 THEN 542	590 GOSUB 727	649 GOSUB 743
533 IF RA=67 THEN 546	591 GOTO 588	650 GOTO 647
534 E=2	592 E=2	651 E=3
535 F=13	593 F=19	652 F=13
536 GOSUB 550	594 GOSUB 716	653 GOSUB 743
537 GOTO 534	595 GOTO 592	654 GOTO 651
538 E=3	596 E=8	655 E=3
539 F=12	597 F=13	656 F=15
540 GOSUB 550	598 GOSUB 727	657 GOSUB 743
541 GOTO 538	599 GOTO 596	658 GOTO 655
542 E=4	600 E=8	659 E=3
543 F=11	601 F=19	660 F=17
544 GOSUB 550	602 GOSUB 716	661 GOSUB 743
545 GOTO 542	603 GOTO 600	662 GOTO 659
546 E=5	604 E=14	663 E=3
547 F=10	605 F=13	664 F=19
548 GOSUB 550	606 GOSUB 727	665 GOSUB 743

666 GOTO 663	722 E=E+1	768 AL=1
667 E=4	723 F=F-2	769 CALL SCREEN(16)
668 F=13	724 NEXT I	770 RETURN
669 GOSUB 743	725 IF AL=1 THEN 777	771 AL=1
670 GOTO 667	726 GOTO 752	772 G2=G2+1
671 E=4	727 FOR I=1 TO 4	773 IF G2>57 THEN 808
672 F=15	728 IF AL=0 THEN 732	774 CALL HCHAR(3,30,G2
673 GOSUB 743	729 CALL HCHAR(E,F,88)	775 CALL SCREEN(10)
674 GOTO 671	730 GOSUB 738	776 RETURN
675 E=4	731 GOTO 733	777 M\$="nouvelle"
676 F=17	732 CALL GCHAR(E,F,A(I	
677 GOSUB 743	733 E=E+1	778 Y=12
678 GOTO 675	734 F=F+1	779 X=22
679 E=4	735 NEXT I	780 GOSUB 790
680 F=19	736 IF AL=1 THEN 777	781 M\$="partie"
681 GOSUB 743	737 GOTO 752	782 Y=14
682 GOTO 679	738 FOR P=0 TO 30 STEP	783 X=23
683 E=5	739 CALL SOUND(1,NO,P)	784 GOSUB 790
684 F=13	740 NEXT P	785 M\$="o n"
685 GOSUB 743	741 NO=NO+200	786 Y=16
686 GOTO 683	742 RETURN	787 X=24
687 E=5	743 FOR I=1 TO 4	788 GOSUB 790
688 F=15	744 IF AL=0 THEN 748	789 GOTO 797
689 GOSUB 743	745 CALL HCHAR(E,F,88)	790 FOR I=9 TO 12
690 GOTO 687	746 GOSUB 738	791 CALL COLOR(I,2,1)
691 E=5	747 GOTO 749	792 NEXT I
692 F=17	748 CALL GCHAR(E,F,A(I	
693 GOSUB 743	749 E=E+6	793 FOR I=1 TO LEN(M\$)
694 GOTO 691	750 NEXT I	794 CALL HCHAR(Y,X+I,A
695 FOR I=1 TO 4	751 IF AL=1 THEN 777	795 NEXT I
696 IF AL=0 THEN 700	752 IF A(1)<>H THEN 75	796 RETURN
697 CALL HCHAR(E,F,88)	753 CALL SOUND(10,3000	797 CALL KEY(3,T,S)
698 GOSUB 738	754 IF A(2)<>H THEN 75	798 IF S=0 THEN 797
699 GOTO 701	755 CALL SOUND(10,3200	799 IF T<>79 THEN 803
700 CALL GCHAR(E,F,A(I	756 IF A(3)<>H THEN 75	800 CALL SCREEN(14)
701 F=F+2	757 CALL SOUND(10,3400	801 CALL CLEAR
702 NEXT I	758 IF A(4)<>H THEN 76	802 GOTO 50
703 IF AL=1 THEN 777	759 CALL SOUND(10,3600	803 CALL CLEAR
704 GOTO 752	760 E=0	804 CALL SCREEN(13)
705 FOR I=1 TO 4	761 F=0	805 FOR I=1 TO 12
706 IF AL=0 THEN 710	762 IF (A(1)=H)*(A(2)=	806 CALL COLOR(I,12,1)
707 CALL HCHAR(E,F,88)	763 GOTO 87	807 NEXT I
708 GOSUB 738	764 IF J=1 THEN 771	808 PRINT TAB(8);"resu
709 GOTO 711	765 G1=G1+1	809 IF G1<>G2 THEN 812
710 CALL GCHAR(E,F,A(I	766 IF G1>57 THEN 808	810 PRINT TAB(10);"mat
711 E=E+1	767 CALL HCHAR(3,3,G1)	811 GOTO 816
712 F=F-1		812 IF G1>G2 THEN 815
713 NEXT I		813 PRINT "etoiles bla
714 IF AL=1 THEN 777		814 GOTO 816
715 GOTO 752		815 PRINT " etoiles r
716 FOR I=1 TO 4		816 FOR DE=1 TO 200
717 IF AL=0 THEN 721		817 NEXT DE
718 CALL HCHAR(E,F,88)		818 END
719 GOSUB 738		
720 GOTO 722		
721 CALL GCHAR(E,F,A(I		



# Gestion de fichiers en assembleur et Basic étendu

Michel Teyssou

**N**ous vous proposons une série de programmes destinés à gérer un fichier d'adresses, et utilisables à partir du Basic étendu, de l'éditeur/assembleur, ou de la Mini-mémoire. Leur mise en œuvre nécessite une imprimante, une extension 32Ko et, bien sûr, un des modules concernés.

Ces programmes offrent les mêmes possibilités que le module "PRK" ou la disquette éditée par Texas Instruments, mais en beaucoup plus performants. En effet, ils autorisent le traitement de 450 fiches, alors que le module "PRK" ne permet de traiter que 102 fiches équivalentes. On peut donc entrer 450 fiches sur une disquette qui contient déjà la totalité des programmes. De plus, la dimension des informations est plus importante qu'avec le module "PRK".

Les enregistrements sont limités à 127 octets, car cela permet, dans la majorité des cas, de loger les informations dont nous avons besoin. De plus, le "Buffer" du PAB emmagasinant à chaque fois 255 octets, cela permet d'avoir en mémoire l'enregistrement demandé plus le suivant, d'où (suivant le type de traitement) un gain de temps appréciable pendant la saisie.

Les saisies sont modifiables ; à cet effet, elles sont regroupées sous forme de "DATA" dans le programme Basic et les zones à modifier sont indiquées dans la version assembleur. En Basic, chaque saisie de chaîne occupe,

en format INTERNAL, sa longueur plus un (octet destiné à renseigner l'ordinateur sur la longueur de la chaîne). En assembleur, les indicateurs ont été supprimés afin d'obtenir plus de place ; de ce fait, un fichier créé avec la version "assembleur" ne sera pas utilisable depuis la version "Basic".

La somme maximum des saisies ne doit en aucun cas dépasser la longueur maximum désignée en FIXED (127 dans le cas présent). L'enregistrement 0 contient les renseignements indispensables :

- TRI (nombre de fiches triées) ;
- TOT (totalité des fiches triées et non triées) ;
- Noms des groupes (8 maxi, 11 lettres au plus chacun) ;
- NG (nombre de groupes, ceci avec quelques variantes d'emplacement en Basic et en assembleur).

Les fiches doivent être regroupées en catégories : secteurs ou groupes (nous avons choisi ce dernier qualificatif, mais on peut prendre n'importe lequel). Si la fiche n'est pas dans un des groupes, elle sera classée "INACTIF" et pourra être effacée lors d'un tri.

Grâce à un tri par indexation, on peut classer très rapidement les noms par ordre alphabétique. Pour donner un ordre d'idée, il faut 45 minutes au module "PRK" pour trier 200 noms, alors qu'il faut moins de cinq minutes au programme "TRI" écrit en Basic et environ trois fois moins à celui écrit en assembleur.

Il faudra se servir souvent de ces

programmes "TRI", car lors de la saisie des noms, la recherche pour vérifier qu'il n'est pas déjà dans le fichier sera extrêmement rapide (méthode des blocs). Sinon, la recherche se fait séquentiellement, dont beaucoup plus lentement.

Dans tous les programmes d'impression ou d'affichage, on peut à tout moment interrompre le processus avec la barre "espace" (idem pour reprendre). On peut arrêter avec <FCTN BACK>. Dans ce cas, il faut maintenir les 2 touches enfoncées jusqu'à l'arrêt.

Dans la totalité des saisies de variables numériques, le fait de rentrer le chiffre zéro (0), permet un retour au dernier menu, et invalide la saisie en cours.

## Programmes Basic

### Conseil de saisie

Lorsque vous rentrerez ces programmes dans votre ordinateur préféré, évitez de mettre les remarques et tête de chapitre ; elles ne sont là que pour clarifier le listing et aucun branchement ne se fait sur ces lignes. De ce fait, votre programme se manipulera plus rapidement au chargement, et tiendra moins de place sur la disquette, (les programmes + "données" et "index" occupent la totalité de la face disquette avec les 450 fiches à condition que les programmes n'aient aucun commentaire). J'ai cherché à les

optimiser au mieux, et du fait de l'imprimante, ils tournent pratiquement aussi vite (pour l'impression) que des programmes similaires écrits en assembleur.

## Sommaire (LOAD)

Programme qui permet de choisir le traitement désiré ; il n'y a rien de spécial à dire sur son fonctionnement qui est hyper classique. Le fait qu'il s'appelle LOAD permet de le charger automatiquement.

### Programme "LOAD"

#### Basic étendu

```
10 |*****
20 |* SOMMAIRE
30 |*****
40 CALL CLEAR :: CALL COLOR(13,8,1)
   :: CALL VCHAR(1,31,129,96) :: C
   ALL SCREEN(8) :: FOR I=1 TO 12 :
   : CALL COLOR(I,16,5) :: NEXT I
50 DISPLAY AT(4,8)BEEP:"SOMMAIRE":
   TAB(8);"-----"
60 DISPLAY AT(8,6):"1 GERER": :TAB
   (6);"2 TRIER": :TAB(6);"3 IMPRI
   MER": :TAB(6);"4 INITIALISER": :
   TAB(6);"5 RECHERCHER"
70 DISPLAY AT(18,6):"6 TITRER EN G
   ROS"
80 CALL KEY(0,K,S) :: IF S=0 THEN 8
   0
90 IF K<49 OR K>54 THEN 80
100 ON K-48 GOTO 110,120,130,140,1
   50,160
110 CALL HCHAR(8,7,62) :: RUN "DSK1
   .GERER"
120 CALL HCHAR(10,7,62) :: RUN "DSK
   1.TRIER"
130 CALL HCHAR(12,7,62) :: RUN "DSK
   1.IMPRIMER"
140 CALL HCHAR(14,7,62) :: RUN "DSK
   1.INITIALISE"
150 CALL HCHAR(16,7,62) :: RUN "DSK
   1.RECHERCHER"
160 CALL HCHAR(18,7,62) :: RUN "DSK
   1.TITRER"
170 END
```

## Initialise le fichier

C'est le premier programme à faire tourner ; il permet d'ouvrir les fichiers, de contrôler si l'espace initial pour les données est suffisant. Vous devrez alors rentrer le nom des groupes dans lesquels vous mettez vos fiches.

### Programme "INITIALISE"

#### Basic étendu

```
10 |*****
20 |INITIALISER LE FICHIER
30 |*****
40 ON WARNING NEXT :: CALL CLEAR :
   : CALL SCREEN(7) :: CALL COLOR(
   13,7,7) :: CALL VCHAR(1,31,129,
   96) :: ON BREAK NEXT
50 FOR A=1 TO 12 :: CALL COLOR(A,5
   ,12) :: NEXT A :: CALL TIT :: C
```

```
ALL ENT :: CALL STOP(A)
60 OPEN #1:"DSK1.",INPUT,INTERNAL
   ,RELATIVE
70 INPUT #1:A$,A,B,C :: CLOSE #1
80 IF C>227 THEN 140
90 CALL NET :: DISPLAY AT(2,1):"LE
   NOMBRE DE SECTEURS SUR LA":"D
   ISQUETTE EST INSUFFISANT..."
-----
100 DISPLAY AT(5,1):" LE FICHIER C
   OMPL ET OCCUPE 227 SECTEURS P
   OUR 450 NOMS -----"
110 DISPLAY AT(10,2):"NOM DU DISQU
   E":" " ;A$ :: " SECTEURS DISP:"
   ;C
120 DISPLAY AT(15,7):"OPTIONS": :
   TAB(6);"1.AUTRE ESSAI": :TAB(6
   );"2.SORTIE"
130 CALL CHOIX(A) :: IF A=1 THEN 60
   ELSE STOP
140 CALL NET :: DISPLAY AT(5,1):"-
   -----":
   LE FICHIER COMPLET OCCUPERA 2
   27 SECTEURS POUR 450 NOMS ----
   -----
150 DISPLAY AT(15,5):"OUVERTURE DU
   FICHIER"
160 CALL ENT :: CALL STOP(A)
170 OPEN #1:"DSK1.DONNEES",RELATIV
   E,INTERNAL,FIXED 127
180 |*****
190 |* Nom de groupes
200 |*****
210 CALL NET
220 DIM G$(8)
230 CALL GROUP :: FOR A=1 TO 8 ::
   DISPLAY AT(9+A,2):A$;" " ;G$(A)
   :: NEXT A
240 NG,A=0
250 CALL NGR :: CALL NUM(A)
260 IF A>8 THEN 340
270 IF G$(A)<>" " AND G$(A)<>" " TH
   EN IF NG THEN NG=NG+1
280 ACCEPT AT(23,5)SIZE(-11):G$(A)
290 IF G$(A)=" " OR G$(A)=" " THEN
   310
300 NG=NG+1 :: IF NG=9 THEN CALL N
   ET2 :: DISPLAY AT(9+A,7):G$(A)
   :: CALL UTIL :: CALL ENT :: CA
   LL STOP(A) :: GOTO 340
310 DISPLAY AT(9+A,7):G$(A)
320 DISPLAY AT(23,2):" ENCORE (O/
   N)? O" :: ACCEPT AT(23,18)VALI
   DATE("ON")SIZE(-1):A$
330 IF A$="O" THEN CALL NET2 :: GO
   TO 250
340 IF NG=0 THEN DISPLAY AT(21,2):
   "AUCUN No DE GROUPE VALIDE" ::
   CALL ENT :: CALL STOP(A) :: CA
   LL NET2 :: GOTO 240
350 CALL NET2 :: DISPLAY AT(21,2):
   "MODIFICATIONS (O/N)?N" :: ACC
   EPT AT(21,22)SIZE(-1)VALIDATE(
   "ON"):A$
360 IF A$="O" THEN CALL NET2 :: GO
   TO 240
370 CALL NET :: DISPLAY AT(14,3):"
   DISPOSITIF D'IMPRESSION ?": :T
   AB(6);"---" :: ACCEPT AT(16,10
   )BEEP SIZE(-16):IMP$
380 DISPLAY AT(14,2)BEEP:"SI.TOUT
   EST OK APPUYEZ SUR": " <EN
   TER> POUR VALIDER": " " <BAC
   K> POUR CORRIGER"
390 CALL STOP(A) :: IF A=15 THEN CA
   LL NET :: GOTO 230
400 TRI$="350" :: TOT$="000" :: TI
   TRE$="O" :: NG$=STR$(NG)
410 PRINT #1,REC 0:TRI$,TOT$,TITRE
   $,IMP$,G$(1),G$(2),G$(3),G$(4)
   ,G$(5),G$(6),G$(7),G$(8),NG$ :
   : CLOSE #1
420 OPEN #3:"DSK1.INDEX",INTERNAL,
   OUTPUT,FIXED 4
430 PRINT #3:" " :: CLOSE #3 :: CA
   LL NET :: DISPLAY AT(9,4)BEEP:
   "*****"
440 DISPLAY AT(11,4)BEEP:"* FICHIE
   R INITIALISE *":TAB(4);**
```

```
*****TAB(4);*****
450 CALL ENT :: CALL STOP(A)
460 END
470 |-----
480 |S/PROGRAMMES FERMES
490 |-----
500 SUB TIT :: DISPLAY AT(9,10):"T
   I 99/4A": : :TAB(4);"INITIALIS
   ER LE FICHIER":TAB(4);"-----
   -----" :: SUBEND
510 SUB ENT :: DISPLAY AT(23,1):"
   Appuyez sur <enter>" :: SU
   BEND
520 SUB GROUP :: DISPLAY AT(5,5):"
   GROUPE SUR FICHIER":TAB(5);"~
   ~~~~~" :: SUBEND
530 SUB NGR :: DISPLAY AT(21,4)BEE
   P:"NOM DU GROUPE No" :: SUBEND
540 SUB NUM(G) :: G=G+1 :: DISPLAY
   AT(21,20):G :: SUBEND
550 SUB UTIL :: DISPLAY AT(21,2):"
   LES 8 No DE GROUPE ONT E
   TES UTILISES" :: SUBEND
560 SUB STOP(K)
570 CALL SOUND(150,800,0)
580 CALL KEY(0,K,S) :: IF NOT S THE
   N 580
590 IF K<>13 AND K<>15 THEN 570
600 SUBEND
610 SUB NET :: CALL VCHAR(1,3,32,6
   72) :: SUBEND
620 SUB NET2 :: FOR A=20 TO 24 ::
   CALL HCHAR(A,3,32,28) :: NEXT A
   :: SUBEND
```

## Gestion du fichier

Ce programme permet de saisir les données et de modifier éventuellement noms de groupes ou informations contenues dans les fiches. Il permet également de marquer une fiche pour l'effacer (avec TRI), de rechercher une fiche ou un groupe et de l'imprimer sur l'écran. Les lignes 1210 à 1230 contiennent les 7 groupes d'informations de la fiche (3 à 9), le nombre à droite correspond au nombre de lettres maxi que contiendra la saisie. En cas de modifications, ce total ne doit en aucun cas dépasser 109 pour les 7 groupes de saisies.

En effet, chaque enregistrement contient 9 informations D\$(1à9), la première occupe 1 octet, soit A (actif), I (inactif), E (effacement). La deuxième occupe 8 octets (groupe dans lequel est la personne), les autres sont ceux de la fiche, soit 1+8+109 = 118+1 octet par information : 18 + 9 = 127.

### Programme "GERER"

#### Basic étendu

```
10 |*****
11 |* GESTION DU FICHIER
12 |*****
20 DIM G$(8),D$(9),XX(9),P$(450)
30 ON WARNING NEXT :: ON BREAK NEX
   T :: CALL CLEAR :: CALL SCREEN(
   7) :: CALL COLOR(13,7,7) :: CALL
```

```

VCHAR(1,31,129,96)
40 FOR A=1 TO 12 :: CALL COLOR(A,5
,12):: NEXT A :: CALL TIT :: CA
LL ENT :: CALL STOP(RG$,A)
50 DISPLAY AT(23,5):"Un instant sv
p...."
60 OPEN #1:"DSK1.",RELATIVE,INTERN
AL,INPUT
70 INPUT #1:A$ :: IF A$="DONNEES"
THEN 100 ELSE IF A$<>" " THEN 70
80 CLOSE #1 :: CALL NET :: CALL RI
EN
90 CALL CHOIX(A):: IF A<1 OR A>2 T
HEN 90 :: IF A=1 THEN 50 ELSE S
TOP
100 CLOSE #1
110 OPEN #1:"DSK1.DONNEES",RELATIV
E,INTERNAL,FIXED 127
120 INPUT #1,REC 0:TRIS,TOT$,TITR
$,IMP$,G$(1),G$(2),G$(3),G$(4)
,G$(5),G$(6),G$(7),G$(8),NG$ :
: TRI=VAL(TRIS):: TOT=VAL(TOT$
):: NG=VAL(NG$)
130 OPEN #3:"DSK1.INDEX",INTERNAL,
INPUT,FIXED 4
140 FOR I=1 TO TRI :: INPUT #3:P$(
I):: NEXT I :: CLOSE #3
150 GROUP$="NNNNNNNN" :: RG$="N"
160 CALL NET :: RESTORE #1 :: CALL
OPT1
170 DISPLAY AT(17,5):"FICHES TRIEE
S =":TRI: TAB(8);"NON TRIEES
=":TOT-TRI
180 CALL CHOIX(A):: IF A<1 OR A>4
THEN 180
190 ON A GOTO 210,720,990,1200
195 *****
200 Mise a jour du fichier
205 *****
210 CALL NET :: CALL OPT2
220 CALL CHOIX(A):: IF A<1 OR A>4
THEN 220
230 IF (A=1)*(TOT>450)=0 THEN 250
240 CALL NET :: CALL PLEIN :: CALL
ENT :: CALL STOP(RG$,A):: GOT
O 210
250 ON A GOTO 270,500,680,160
260 ***** Ajouter *****
270 CALL NET :: CALL NOM :: CALL A
JOUT :: GOSUB 1450 :: IF N$="
" OR N$="" THEN
480 ELSE IF A=0 THEN 340
280 CALL DEJA :: IF D$(1)="E" THEN
CALL EFF ELSE IF D$(1)="I" TH
EN CALL INAC :: CALL AUTRE ELS
E CALL AUTRE
290 ACCEPT AT(24,24)VALIDATE("ON")
SIZE(-1):A$ :: IF A$="N" THEN
270
300 REP=Z :: IF D$(1)<>"E" THEN 34
0
310 IF D$(2)=GROUP$ THEN D$(1)="I"
GOTO 460
320 D$(1)="A" :: GOTO 460
330 RESTORE 1220 :: CALL NET :: CA
LL DONNE :: CALL MOD
340 J=409 :: GOSUB 1400 :: D$(3)=N
$
350 DISPLAY AT(24,2)BEEP:"C'EST CO
RRECT? (O/N) O"
360 ACCEPT AT(24,23)VALIDATE("ON")
SIZE(-1):A$ :: IF A$="N" THEN
RESTORE 1220 :: DISPLAY AT(23,
1):: GOTO 340
370 D$(2)=GROUP$ :: IF NG=0 THEN D
$(1)="I" :: GOTO 450
380 CALL NET :: CALL NOU :: DISPLA
Y AT(9,1):"-----
:":TAB(2);D$(8);" "":D
$(3):"-----
:":
390 FOR A=1 TO NG
400 DISPLAY AT(12+A,5):G$(A);TAB(1
7);"O/N N" :: ACCEPT AT(12+A,2
1)VALIDATE("ON")SIZE(-1)BEEP:A
$
410 GOSUB 1510
420 NEXT A
430 DISPLAY AT(24,3):"C'EST CORREC
T? (O/N) O" :: ACCEPT AT(24,24
)VALIDATE("ON")SIZE(-1):A$ ::

```

```

IF A$="N" THEN D$(2)=GROUP$ ::
GOTO 380
440 IF D$(2)=GROUP$ THEN D$(1)="I"
ELSE D$(1)="A"
450 IF REP=0 THEN Z=TOT+1 ELSE Z=R
EP
460 GOSUB 1280 :: IF REP=0 THEN TO
T=TOT+1
470 GOSUB 1250
480 CALL CONT :: ACCEPT AT(24,24)V
ALIDATE("ON")SIZE(-1):A$ :: IF
A$="N" THEN 160 ELSE 270
490 ***** Modifier *****
500 CALL NET :: CALL NOM :: CALL M
OD :: GOSUB 1450
510 RG$="N" :: IF A=0 THEN CALL NE
T :: CALL PAS :: CALL ENT :: C
ALL STOP(RG$,A):: GOTO 160
520 CALL NET :: RESTORE 1210
530 FOR A=3 TO 9 :: READ A$,XX(A)
540 DISPLAY AT(2+Z,1):A$;" "":SEG$
(A$,1,POS(A$,".",1)-1):D$(A)
550 ZZ=ZZ+3 :: NEXT A
560 ZZ=0 :: CALL LIGNE
570 ACCEPT AT(24,23)VALIDATE(DIGIT
)SIZE(1):B :: IF B=0 THEN 660
ELSE IF B=1 THEN 590
580 CALL DON :: ACCEPT AT(24,2)SIZ
E(-XX(B))BEEP:A$ :: D$(B)=A$ :
: GOTO 520
590 CALL NET :: W=1 :: CALL GR ::
GOSUB 1480 :: CALL CH2
600 ACCEPT AT(24,19)VALIDATE("0123
")SIZE(1)BEEP:B :: IF B=0 THEN
660 ELSE IF B=1 THEN 650
610 IF B=2 THEN CALL AJN ELSE CALL
EFG
620 ACCEPT AT(24,26)SIZE(1)VALIDAT
E("12345678"):A
630 IF B=2 THEN A$="O" ELSE A$="N"
:: GOSUB 1510 :: GOTO 590
640 GOSUB 1510 :: GOTO 590
650 GOSUB 1280
660 CALL MISE :: ACCEPT AT(24,28)S
IZE(-1)VALIDATE("ON")BEEP:A$ :
: IF A$="N" THEN 160 ELSE 210
670 ***** Effacer *****
680 CALL NET :: CALL NOM :: CALL E
FFA :: GOSUB 1450 :: IF A=0 TH
EN CALL NET :: CALL PAS :: CAL
L ENT :: CALL STOP(RG$,A):: GO
TO 160
690 D$(1)="E" :: GOSUB 1280 :: CAL
L AEFF :: ACCEPT AT(24,27)VALI
DATE("ON")SIZE(-1):A$
700 IF A$="O" THEN 680 ELSE 160
705 *****
710 l' Nom de groupes
715 *****
720 CALL NET :: CALL OPT4 :: CALL
CHOIX(A)
730 IF A<1 OR A>3 THEN 730
740 ON A GOTO 760,910,160
750 !** Ajouter un groupe **
760 IF NG=8 THEN CALL UTIL :: CALL
ENT :: CALL STOP(RG$,A):: GOT
O 160
770 CALL NET :: W=0 :: CALL GROUP
:: GOSUB 1480 :: CALL NOUV
780 ACCEPT AT(22,26)VALIDATE(DIGIT
)SIZE(1)BEEP:A$ :: IF A=0 THEN
720
790 IF A<1 OR A>8 THEN 780
800 Y=1 :: IF G$(A)="" OR G$(A)="
" THEN 840
810 CALL REM :: ACCEPT AT(24,27)V
ALIDATE("ON")SIZE(-1):A$
820 IF A$="N" THEN CALL CONT :: GO
TO 890
830 Y=0 :: GOTO 850
840 NG=NG+1
850 CALL NGR :: DISPLAY AT(24,1)::
ACCEPT AT(24,2)SIZE(-1):G$(A)
860 IF (G$(A)="" + (G$(A)=" " ) THEN
NG=NG-1
870 GOSUB 1250 :: IF Y<>1 THEN GOS
UB 1660
880 CALL NET :: CALL CONT
890 ACCEPT AT(24,24)VALIDATE("ON")
SIZE(-1):A$ :: IF A$="N" THEN

```

```

160 ELSE 760
900 l' Effacer un groupe **
910 IF NG=0 THEN CALL NET :: CALL
PASG :: CALL ENT :: CALL STOP(
RG$,A):: GOTO 720
920 CALL NET :: W=0 :: CALL GROUP
:: GOSUB 1480 :: CALL EFG :: A
CCEPT AT(24,24)VALIDATE(DIGIT
)SIZE(1):A
930 IF A=0 THEN 720 ELSE IF A>8 TH
EN 920
940 IF G$(A)="" OR G$(A)=" " THEN
960
950 G$(A)="" :: NG=NG-1 :: GOSUB 1
250 :: GOSUB 1660 :: CALL NET
960 CALL AEFF :: ACCEPT AT(24,27)V
ALIDATE("ON")SIZE(-1):A$
970 IF A$="O" THEN 910 ELSE 160
975 *****
980 l' Recherche
985 *****
990 CALL NET :: CALL OPT3 :: CALL
CHOIX(A):: IF A<1 OR A>4 THEN
990 :: X=0
1000 ON A GOTO 1020,1060,1110,160
1010 ***** Une fiche *****
1020 CALL NET :: CALL NOM :: CALL
AFF :: GOSUB 1450
1030 IF A=0 THEN CALL NET :: CALL
PAS :: CALL ENT :: CALL STOP(
RG$,A):: GOTO 160
1040 GOSUB 1560 :: IF A=13 THEN DI
SPAY AT(24,1):: CALL ATR ::
ACCEPT AT(23,24)VALIDATE("ON"
)SIZE(-1)BEEP:A$ :: IF A$="O"
THEN 990 ELSE 160
1050 ***** Un groupe *****
1060 IF NG=0 THEN CALL NET :: CALL
PAS :: CALL ENT :: CALL STOP
(RG$,A):: GOTO 160
1070 CALL NET :: W=0 :: CALL GROUP
:: GOSUB 1480 :: CALL GAF
1080 ACCEPT AT(24,24)VALIDATE(DIGI
T)SIZE(-1):Y :: IF Y=0 THEN 9
90 ELSE IF Y>8 THEN 1080
1090 IF G$(Y)="" OR G$(Y)=" " THEN
1080 ELSE 1120
1100 l' Toutes les fiches *
1110 X=1
1120 FOR Z=1 TO TOT
1130 GOSUB 1270
1140 IF X<>1 THEN IF SEG$(D$(2),Y,
1)="" THEN 1160
1150 GOSUB 1560
1160 NEXT Z
1170 CALL NET :: CALL FIN :: CALL
ENT :: CALL STOP(RG$,A)
1180 IF A=13 THEN CALL ATR :: ACCE
PT AT(23,24)VALIDATE("ON")SIZ
E(-1):A$ :: IF A$="O" THEN 99
0
1190 GOTO 160
1200 CLOSE #1 :: STOP
1201 |-----|
1202 | ZONE DES DATAS
1203 |-----|
1210 DATA NOM PRENOM.....
,23
1220 DATA ADRESSE.....
...,27,CODE POSTAL.,5,VILLE..
.....,15
1230 DATA NO TEL.....,12,TITRE.,
5,DIVERS.....,22
1240 DATA INACTIF,EFFACEMENT
1242 |-----|
1243 | ZONE S/P OUVERTS
1244 |-----|
1250 TRIS=STR$(TRI):: TOT$=STR$(TO
T):: NG$=STR$(NG)
1255 PRINT #1,REC 0:TRIS,TOT$,TITR
E$,IMP$,G$(1),G$(2),G$(3),G$(
4),G$(5),G$(6),G$(7),G$(8),NG
$ :: RETURN
1260 INPUT #1,REC Z:D$(1),D$(2),D$
(3):: RETURN
1270 INPUT #1,REC Z:D$(1),D$(2),D$
(3),D$(4),D$(5),D$(6),D$(7),D
$(8),D$(9):: RETURN
1280 PRINT #1,REC Z:D$(1),D$(2),D$
(3),D$(4),D$(5),D$(6),D$(7),D

```

```

$ (8),D$ (9):: RETURN
1290 A=0 :: IF TRI=0 THEN 1360 ELS
E IF TOT THEN I=1 :: J=TRI EL
SE RETURN
1300 IF D$ (3)=" " OR D$ (3)=" " THEN
RETURN
1310 ZT=INT((I+J)/2):: Z=VAL(P$(ZT
)): GOSUB 1260
1320 IF D$ (3)=N$ THEN A=Z :: GOSUB
1270 :: RETURN
1330 IF D$ (3)<N$ THEN I=ZT+1 ELSE
J=ZT-1
1340 IF (ZT<1)+(I>J)=0 THEN 1310
1350 IF TRI=TOT THEN RETURN
1360 FOR Z=TRI+1 TO TOT
1370 GOSUB 1260
1380 IF D$ (3)=N$ THEN A=Z :: GOSUB
1270 :: RETURN
1390 NEXT Z :: Z=0 :: RETURN
1400 A=INT(J/100)
1410 FOR A=A TO J-A*100 :: READ A$
,I
1420 DISPLAY AT(16,2):A$: "?" ::
CALL SOUND(150,400,10)
1430 ACCEPT AT(18,2) SIZE(-I):D$ (A)
1440 NEXT A :: RETURN
1450 J=303 :: RESTORE 1210
1460 GOSUB 1400 :: N$=D$ (3)
1470 GOSUB 1290 :: RETURN
1480 FOR A=1 TO 8 :: DISPLAY AT(7+
A,2):A;TAB(6);" " ;G$(A):: IF
W=0 OR SEG$(D$ (2),A,1)="N" T
HEN 1500
1490 DISPLAY AT(7+A,23):""
1500 NEXT A :: RETURN
1510 IF A=1 THEN D$ (2)=A$&SEG$(D$ (
2),2,7):: GOTO 1540
1520 IF A=8 THEN D$ (2)=SEG$(D$ (2),
1,7)&A$ :: GOTO 1540
1530 D$ (2)=SEG$(D$ (2),1,A-1)&A$&SE
G$(D$ (2),A+1,8-A)
1540 IF D$ (1)="E" THEN RETURN ELSE
IF D$ (2)=GROUP$ THEN D$ (1)="
I" :: RETURN ELSE D$ (1)="A" :
: RETURN
1560 CALL NET :: DISPLAY AT(2,1):D
$(8):D$ (3): :D$ (4):D$ (5);" ";
D$ (6):D$ (7):D$ (9)
1570 C=0 :: IF D$ (1)="I" THEN 1610
1580 FOR A=1 TO NG :: IF SEG$(D$ (2
),A,1)="N" THEN 1600
1590 DISPLAY AT(14+C,18):G$(A):: C
=C+1
1600 NEXT A
1610 IF D$ (1)="A" THEN 1650
1620 RESTORE 1240
1630 IF D$ (1)="I" THEN READ A$ ELS
E READ A$ :: READ A$
1640 DISPLAY AT(1,18):A$
1650 CALL BACK :: RG$="O" :: CALL
STOP(RG$,A):: RG$="N" :: IF A
=13 THEN RETURN ELSE Z=TOT+1
:: RETURN
1660 IF TOT THEN CALL NET :: CALL
MOM :: A$="N" ELSE RETURN
1670 FOR Z=1 TO TOT :: GOSUB 1270
:: IF D$ (1)="E" OR SEG$(D$ (2
),A,1)=A$ THEN 1690
1680 GOSUB 1510 :: GOSUB 1280
1690 NEXT Z :: RETURN
1695 |-----|
1696 ! ZONE S/P FERMES
1697 |-----|
1700 SUB BACK :: DISPLAY AT(24,7):
"<ENTER OU BACK>" :: SUBEND
1710 SUB OPT4 :: DISPLAY AT(5,7):"
NOMS DE GROUPE":TAB(7);"~~~~~
~~~~~": :TAB(6);"1.AJ
OUTER UN NOM"
1720 DISPLAY AT(11,5):TAB(6);"2.EF
FACER UN NOM": :TAB(6);"3.RET
OUR" :: SUBEND
1730 SUB TIT :: DISPLAY AT(9,11):"
TI 99/4A": :TAB(6);"GESTION
DU FICHIER":TAB(6);"~~~~~
~~~~~": :SUBEND
1740 SUB ENT :: DISPLAY AT(23,5):"
Appuyez sur <enter>" :: SUBEN
D
1750 SUB OPT1 :: DISPLAY AT(3,5):"
OPTIONS DU FICHIER":TAB(5);"~

```

```

~~~~~": :TAB(5)
;"1.MISE A JOUR": :TAB(5);"2.
NOMS DE GROUPE"
1760 DISPLAY AT(11,5):"3.RECHERCHE
": :TAB(5);"4.SORTIE" :: SUBE
ND
1770 SUB CHOIX(A):: DISPLAY AT(23,
6)BEEP:"VOTRE CHOIX ?" :: ACC
EPT AT(23,20) SIZE(1):A :: SUB
END
1780 SUB RIEN :: DISPLAY AT(7,2):"
LES DONNEES NE SONT PAS": "
SUR CETTE DISQUETTE !": :
:TAB(9);"OPTIONS":TAB(9);"~~~~~
~~~~~"
1790 DISPLAY AT(17,7):"1.AUTRE ESS
AI": :TAB(7);"2.SORTIE" :: SU
BEND
1800 SUB OPT3 :: DISPLAY AT(5,8):"
RECHERCHE":TAB(8);"~~~~~
~~~~~": :TAB(7);"1.UNE FICHE": :T
AB(7);"2.UN GROUPE"
1810 DISPLAY AT(13,7):"3.TOUT LE F
ICHIER": :TAB(7);"4.RETOUR" :
:SUBEND
1820 SUB OPT2 :: DISPLAY AT(5,8):"
MISE A JOUR":TAB(8);"~~~~~
~~~~~": :TAB(8);"1.AJOUTER": :T
AB(8);"2.MODIFIER"
1830 DISPLAY AT(12,8):"3.EFFACER":
:TAB(8);"4.RETOUR" :: SUBEND
1840 SUB TITRE :: DISPLAY AT(16,2)
:"TITRE": :?" :: SUBEND
1850 SUB DONNE :: DISPLAY AT(4,2):
"RENTREZ LES DONNEES": :QUE
VOUS VOULEZ" :: SUBEND
1860 SUB NOM :: DISPLAY AT(4,2):"R
ENTREZ LE NOM DE LA": :PERS
ONNE QUE VOUS VOULEZ" :: SUBE
ND
1870 SUB AJOUT :: DISPLAY AT(8,2):
"AJOUTER." :: SUBEND
1880 SUB MOD :: DISPLAY AT(8,2):"M
ODIFIER." :: SUBEND
1890 SUB EFFA :: DISPLAY AT(8,2):"
EFFACER." :: SUBEND
1900 SUB AFF :: DISPLAY AT(8,2):"A
FFICHER." :: SUBEND
1910 SUB GROUP :: DISPLAY AT(5,5):
"GROUPE SUR FICHIER":TAB(5);
"~~~~~
~~~~~": :SUBE
ND
1920 SUB NOU :: DISPLAY AT(2,4):"I
NDIQUEZ PAR (O/N) LE": :GRO
UPE DANS LEQUEL METTRE": "
LE NOUVEAU NOM." :: SUBEND
1930 SUB CONT :: DISPLAY AT(24,1):
"VOUS CONTINUEZ (O/N)?O" ::
SUBEND
1940 SUB MISE :: DISPLAY AT(24,1):
"UNE AUTRE MISE A JOUR(O/N)?O
" :: SUBEND
1950 SUB AEFF :: DISPLAY AT(24,2)B
EEP:"AUTRE MODIFICATION (O/N)
?O" :: SUBEND
1960 SUB PLEIN :: DISPLAY AT(12,2)
:"DESOLE LE FICHIER EST PLEIN
" :: SUBEND
1970 SUB PAS :: DISPLAY AT(12,3):"
CETTE PERSONNE N'EST PAS": :
SUR FICHIER" :: SUBEND
1980 SUB DEJA :: DISPLAY AT(22,2):
"PERSONNE DEJA SUR FICHIER":
:SUBEND
1990 SUB AUTRE :: DISPLAY AT(24,2)
BEEP:"CHANGEMENT (O/N)? N
" :: SUBEND
2000 SUB EFF :: DISPLAY AT(23,2)B
EEP:"ET MARQUEE EFFACEMENT":
REINTRODUCTION (O/N)? " :: SUB
END
2010 SUB INAC :: DISPLAY AT(23,2)B
EEP:"ET MARQUEE INACTIVE" ::
SUBEND
2020 SUB GR :: DISPLAY AT(2,1):"
GROUPE MEMBRE" ::
SUBEND
2030 SUB CH2 :: DISPLAY AT(20,1):"
1 POUR INCHANGER": " 2 POUR
AJOUTER A UN GROUPE": " 3 POU
R EFFACER D'UN GROUPE": "
VOTRE CHOIX ?" :: SUBEND

```

```

2040 SUB AJN :: DISPLAY AT(24,2):"
AJOUTER A QUEL NO ?" :: SUB
END
2050 SUB PASG :: DISPLAY AT(12,2):
"AUCUN GROUPE SUR FICHIER":
"~~~~~
~~~~~": :
SUBEND
2060 SUB NOUV :: DISPLAY AT(22,6):
"NOUVEAU GROUPE No ?" :: SUBE
ND
2070 SUB REMP :: DISPLAY AT(24,2):
"REMPLACEMENT GROUPE O/N? O"
:: SUBEND
2080 SUB NGR :: DISPLAY AT(22,2):"
NOM DU NOUVEAU GROUPE ?": "
" :: SUBEND
2090 SUB UTIL :: DISPLAY AT(23,2):
"LES 8 GROUPE SONT UTILISES"
:: SUBEND
2100 SUB EFG :: DISPLAY AT(24,2):"
NO GROUPE A EFFACER ?" :: SUB
END
2110 SUB MOM :: DISPLAY AT(12,2):"
MODIFICATION EN COURS": : " UN
MOMENT SVP..." :: SUBEND
2120 SUB LIGNE :: DISPLAY AT(23,2)
BEEP:"NO DE LA LIGNE A CHANGE
R": " 1 POUR INCHANGER ?" ::
SUBEND
2130 SUB DON :: DISPLAY AT(23,2):"
NOUVELLE DONNEE": "?" :: SUBEN
D
2140 SUB GAF :: DISPLAY AT(24,1):"
GROUPE A AFFICHER ?" :: SU
BEND
2150 SUB FIN :: DISPLAY AT(12,7):"
FIN DU FICHIER":TAB(7);"~~~~~
~~~~~": :SUBEND
2160 SUB ATR :: DISPLAY AT(23,2):"
AUTRE RECHERCHE? (O/N)N" :: S
UBEND
2170 SUB STOP(RG$,A)
2180 CALL SOUND(150,800,0)
2190 CALL KEY(0,A,B):: IF NOT ■ TH
EN 2190 :: IF A<>13 AND(A=15)
*(RG$="O")=0 THEN 2180
2200 SUBEND
2210 SUB NET :: CALL VCHAR(1,3,32,
672):: SUBEND

```

## Trier le fichier

Aussitôt que vous avez une dizaine de fiches, classez-les par ordre alphabétique. Ce programme va créer un index qui permettra aux autres programmes d'aller chercher les informations alphabétiquement et vous pourrez, en option, effacer des fiches. Utilisez-le fréquemment, il est en plus assez rapide.

### Programme "TRIER"

#### Basic Etendu

```

10 |*****|
20 |TRI ALPHABETIQUE FICHIER
30 |*****|
40 DIM G$(8),D$(9),P$(450),S$(450)
50 CALL CLEAR :: CALL COLOR(13,7,7
):: CALL SCREEN(7)
60 CALL VCHAR(1,31,128,96):: FOR A
=1 TO 12 :: CALL COLOR(A,5,12):
: NEXT A
70 CALL TITRE :: CALL ENT :: CALL
STOP
80 DISPLAY AT(23,6):"UN INSTANT SV
P...."
90 OPEN #1:"DSK1.",INPUT,RELATIVE
,INTERNAL
100 INPUT #1:T$: IF T$="DONNEES"
THEN 140 ELSE IF T$<>" " THEN
100

```



```

110 CLOSE #1 :: CALL NET :: CALL R
    IEN
120 ACCEPT AT(23,22)VALIDATE("12")
    :T
130 IF T=1 THEN 80 ELSE END
140 CLOSE #1
150 OPEN #1:"DSK1.DONNEES",RELATIV
    E,INTERNAL,FIXED 127
160 INPUT #1:TRI$,TOT$,TITRE$,IMP$,
    G$(1),G$(2),G$(3),G$(4),G$(5),
    G$(6),G$(7),G$(8),NG$: : TRI=
    VAL(TRI$): : TOT=VAL(TOT$)
170 CALL NET
180 IF TOT=0 THEN DISPLAY AT(16,2)
    : "PAS DE DONNEES SUR FICHIER":
    : *****
    : : CLOSE #1 :: CALL ENT :: CAL
    L STOP :: END
190 DISPLAY AT(6,8): "OPTIONS DE TR
    I":TAB(8); "~~~~~"
200 DISPLAY AT(10,2): "1-SUPPRIMER
    LES EFFACEMENTS": : : " 2-EFFAC
    EMENTS ET INACTIFS": : : " 3-AU
    CUNE SUPPRESSION"
210 DISPLAY AT(22,6)BEEP:"VOTRE CH
    OIX ?" :: ACCEPT AT(22,21)SIZE
    (1)VALIDATE("123"):Y :: IF Y=3
    THEN GOSUB 660 :: GOTO 400
220 |*****
230 |*Routine d'effacement
240 |*****
250 CALL NET :: CALL OP :: CALL DE
    P :: CALL DON(TOT)
260 DISP,C=0
270 C=C+1 :: IF C>TOT THEN GOSUB 6
    60 :: GOTO 400
280 Z=C :: GOSUB 670
290 IF D$(1)="E" OR(D$(1)="I" AND
    Y=2)THEN 310
300 IF DISP THEN Z=C-DISP :: GOSUB
    690 :: GOTO 270 ELSE 270
310 IF TOT>TRI OR C<>TOT THEN 340
320 DISP=DISP+1
330 IF C<=TRI THEN TRI=TRI-1 ELSE
    270
340 Z=TOT :: GOSUB 680 :: TOT=TOT-
    1
350 IF D$(1)="E" OR(D$(1)="I" AND
    Y=2)THEN 310
360 Z=C :: GOSUB 690 :: GOTO 270
370 |*****
380 |* Routine de triage
390 |*****
400 CALL NET :: CALL OP :: CALL TR
    I :: CALL DON(TOT)
410 FOR Z=1 TO TOT
420 INPUT #1,REC Z:D$(1),D$(2),D$(
    3)
430 S$(Z)=SEG$(D$(3),1,4)&SEG$(D$(
    3),POS(D$(3)," ",1)+1,1):: P$(
    Z)=STR$(Z)
440 NEXT Z :: G=1
450 G=3*G+1 :: IF G<TOT THEN 450
460 G=INT(G/3)
470 FOR I=G TO TOT :: P1$=P$(I)::
    S1$=S$(I)
480 FOR J=I-G TO 1 STEP -G :: P2$=
    P$(J):: S2$=S$(J)
490 IF S1$>S2$ THEN 510
500 P$(J+G)=P2$ :: S$(J+G)=S2$ ::
    NEXT J
510 P$(J+G)=P1$ :: S$(J+G)=S1$
520 NEXT I
530 IF G>1 THEN 460
540 |*****
550 |*Enregistrement index
560 |*****
570 CLOSE #1
580 OPEN #3:"DSK1.INDEX",OUTPUT,IN
    TERNAL,FIXED 4
590 FOR I=1 TO TOT :: PRINT #3:P$(
    I):: NEXT I
600 CLOSE #3
610 DISPLAY AT(23,2): "INDEX ENREGI
    STRE. <ENTER>"
620 CALL STOP :: END
630 |*****
640 |* Sous programmes
650 |*****
660 TOT$=STR$(TOT):: PRINT #1,REC

```

```

0:TOT$,TOT$,TITRE$,IMP$,G$(1),
G$(2),G$(3),G$(4),G$(5),G$(6),
G$(7),G$(8),NG$: : RETURN
670 INPUT #1,REC Z:D$(1):: RETURN
680 INPUT #1,REC Z:D$(1),D$(2),D$(
    3),D$(4),D$(5),D$(6),D$(7),D$(
    8),D$(9):: RETURN
690 PRINT #1,REC Z:D$(1),D$(2),D$(
    3),D$(4),D$(5),D$(6),D$(7),D$(
    8),D$(9):: RETURN
700 SUB TITRE :: DISPLAY AT(9,10):
    "TI 99/4A": :TAB(5);" TRI ALPH
    ABETIQUE " : :TAB(6);" PAR IND
    EXATION " : : SUBEND
710 SUB ENT :: DISPLAY AT(23,6): "A
    PPUEZ SUR ENTER" :: SUBEND
720 SUB RIEN :: DISPLAY AT(9,1)BEE
    P:" PAS DE FICHIER SUR DISQUE"
    : "~~~~~"
    : :TAB(7);"OPTIONS": : : " 1.
    NOUVEL ESSAI": : " 2. SORTIE"
730 DISPLAY AT(23,7): "VOTRE CHOIX
    ?" :: SUBEND
740 SUB OP :: DISPLAY AT(8,5): "OPE
    RATION EN COURS":TAB(5);"~~~~~"
    : : SUBEND
750 SUB STOP
760 CALL SOUND(150,400,0)
770 CALL KEY(0,A,B):: IF NOT B THE
    N 770 :: IF A<13 THEN 760
780 SUBEND
790 SUB NET :: CALL VCHAR(1,3,32,6
    72):: SUBEND
800 SUB DON(TOT):: CALL SOUND(150,
    800,0):: DISPLAY AT(18,3): "LE NB D
    ES DONNEES VA DE": : "
    ";1;" A ";TOT :: SUBEND
810 SUB DEP :: DISPLAY AT(16,10): "
    AU DEPART" :: SUBEND
820 SUB TRI :: DISPLAY AT(16,9): "P
    OUR LE TRI" :: SUBEND

```

Vous pouvez imprimer une fiche, un groupe de fiches ou la totalité du fichier. Vous pouvez imprimer un fichier organisé sous forme de colonnes. Le programme utilise une imprimante avec des caractères condensés (136/ligne), cela permet d'afficher la totalité des informations sur une ligne. Dans le cas d'une imprimante sans caractères condensés, il faudra modifier les lignes 490 à 500, 610 et 660, puis afficher sur 2 lignes chaque fiche ou bien se restreindre. Vous pouvez imprimer des étiquettes d'adresses. De nombreux paramètres vous permettront de centrer votre étiquette au mieux. J'utilise pour ma part, deux étiquettes de front (cas le plus courant) et les paramètres que j'utilise sont stockés à la ligne 110. Faites de même quand vous aurez déterminé les vôtres. Une étiquette modèle est imprimée pour déterminer le format maxi de vos données. Toutes les spécifications de l'imprimante qui sont employées dans les

## Imprimer le fichier

### Variables utilisées

#### Enregistrement 0

#### Nombre d'octets

TRI\$	Enregistrements triés	3 + 1
TOT\$	Nb total de fiches	3 + 1
TITRE\$	Option titre	1 + 1
IMP\$	Nom de l'imprimante	16 + 1
D\$(1 à 8)	noms des 8 groupes	88 + 8
NG\$	Nb de groupes utilisés	3 + 1

Les variables numériques TRI, TOT, NG, sont stockées en tant que chaîne pour qu'elles n'occupent que 3 octets maximum (nb de 3 chiffres), au lieu de 8 pour une variable numérique.

#### Autres enregistrements

D\$(1)	A (actif) ou I (inactif) ou E (effacement)
D\$(2)	8 "N" ou "O" selon que la personne se situe ou non dans un des 8 groupes. Par exemple "ONONNNNN" pour une personne faisant partie des groupes 1 et 8.
*D\$(3)	Nom, Prénom
D\$(4)	Adresse
D\$(5)	Code postal
D\$(6)	Ville
D\$(7)	Titre
D\$(8)	Divers

différents programmes sont expliquées dans le programme **TITRER EN GROS**.

## Programme "IMPRIMER"

### Basic étendu

```

5 |-----
6 | IMPRIMER LE FICHIER
7 |-----
10 ON WARNING NEXT :: OPTION BASE
11 1 :: ON BREAK NEXT
20 DIM G$(8),D$(9),ETIQ$(4,3),P$(4
50)
30 INACTIF$="N" :: BLANC$=RPT$(" ",
40)
40 CALL CLEAR :: CALL SCREEN(7)
50 CALL COLOR(13,7,7)
60 CALL VCHAR(1,31,129,96)
70 FOR I=1 TO 12
80 CALL COLOR(I,5,12)
90 NEXT I
100 CALL TITRE :: CALL ENT
110 NCOL=80 :: LARG=36 :: ESP=5 ::
NBET=2 :: DEP,NREPS,RECNO,ENR
=1
120 CALL STOP
130 DISPLAY AT(23,6):"UN INSTANT S
VP...."
140 OPEN #1:"DSK1.",INTERNAL,RELAT
IVE,INPUT
150 INPUT #1:T$ :: IF T$="DONNEES"
THEN 190 ELSE IF T$<>" " THEN
150
160 CLOSE #1 :: CALL NET :: CALL R
IEN
170 DISPLAY AT(23,6):"VOTRE CHOIX
?" :: ACCEPT AT(23,20)VALIDATE
("12")SIZE(1):T
180 IF T=2 THEN 900 ELSE 130
190 CLOSE #1
200 GOSUB 920 :: GOSUB 1850
210 CALL NET :: CALL DISP(IMP$)
220 ACCEPT AT(14,22)VALIDATE("ON")
SIZE(-1):A$
230 IF A$="N" THEN 250
240 RESTORE 2020 :: READ A$ :: INP
UT A$:IMP$ :: GOSUB 940
250 CALL NET :: CALL OPT
260 DISPLAY AT(23,6)BEEP:"VOTRE CH
OIX ?" :: ACCEPT AT(23,20)SIZE
(1)VALIDATE("1234"):A
270 ON A GOTO 290,450,710,900
275 |-----
280 | Toute information
285 |-----
290 OPEN #9:IMP$,OUTPUT :: GOSUB 1
450
300 CALL NET :: CALL IMP
310 PRINT #9:" FICHIER D'ADRESSES
": " -----":
320 FOR RECNO=1 TO TOT :: GOSUB 97
0
330 IF M<12 THEN 410
340 IF TITRES="O" THEN PRINT #9:D$(
8)
350 PRINT #9:D$(3):D$(4):D$(5);" "
:D$(6):D$(7):D$(9)
360 PRINT #9
370 FOR I=1 TO 8 :: IF SEG$(D$(2),
I,1)="N" THEN 390
380 PRINT #9:TAB(5);G$(I)
390 NEXT I
400 PRINT #9
410 NEXT RECNO
420 GOSUB 1620
430 CLOSE #9 :: GOTO 250
435 |-----
440 | Fichier organise
445 |-----
450 OPEN #9:IMP$&".CR",OUTPUT :: G
OSUB 1450
460 INPUT "TITRE ET DATE: ":DA$ ::
CALL NET :: CALL IMP
470 PRINT #9:DA$;CHR$(10);CHR$(10)
480 RESTORE 2030 :: READ A$,A1$,A2
$,A3$,A4$
490 PRINT #9:CHR$(27);"C";CHR$(16)
; "004";A$;
500 PRINT #9:CHR$(16);"033";A1$;CH
R$(16);"066";A2$;CHR$(16);"090
";A3$;CHR$(16);"105";A4$;CHR$(
10)
510 READ A$,A1$,A2$,A3$,A4$
520 PRINT #9:CHR$(16);"004";A$;
530 PRINT #9:CHR$(16);"033";A1$;CH
R$(16);"066";A2$;CHR$(16);"090
";A3$;CHR$(16);"105";A4$;CHR$(
10)
540 DISPLAY AT(20,2):"VERIFIEZ L'A
LIGNEMENT": "
<ENTER>"
550 ACCEPT AT(22,24):A$ :: DISPLAY
AT(20,2) :: DISPLAY AT(22,10):
: A=0
560 FOR I=1 TO TRI :: RECNO=VAL(P$(
I))
570 PRINT #9:CHR$(24):: GOSUB 970
580 IF M<12 THEN 620
590 A=A+1
600 PRINT #9,USING "### ":A:: PRI
NT #9:D$(3);
610 PRINT #9:CHR$(16);"033";D$(4);
CHR$(16);"066";D$(5);" ";D$(
6);CHR$(16);"090";D$(7);CHR$(1
6);"105";D$(9);CHR$(10)
620 NEXT I
630 IF TOT=TRI THEN 680
640 FOR RECNO=TRI+1 TO TOT :: PRIN
T #9:CHR$(24):: GOSUB 970 :: I
F M<12 THEN 670 :: A=A+1
650 PRINT #9,USING "### ":A:: PRI
NT #9:D$(3);
660 PRINT #9:CHR$(16);"033";D$(4);
CHR$(16);"066";D$(5);" ";D$(
6);CHR$(16);"090";D$(7);CHR$(1
6);"105";D$(9);CHR$(10)
670 NEXT RECNO
680 GOSUB 1620
690 A=0 :: PRINT #9:CHR$(27);"N" :
: CLOSE #9 :: GOTO 250
695 |-----
700 | Impression d'etiquette
705 |-----
710 OPEN #9:IMP$,OUTPUT :: GOSUB 1
450
720 RESTORE 1910 :: CALL NET
730 DISPLAY AT(3,1):"-----"
"-----": " OPTIONS DE
MISE EN PLACE": " DES ETI
QUETTES": "-----"
740 DISPLAY AT(11,1)BEEP:" CARACTE
RES GRAS (O/N)? O" :: ACCEPT A
T(11,25)VALIDATE("ON")SIZE(-1)
:A$ :: IF A$="O" THEN EP$="B"
ELSE EP$="N"
750 MOINS=32 :: PLUS=132 :: VAR=NC
OL :: GOSUB 1770 :: NCOL=VAR
760 MOINS=1 :: PLUS=4 :: VAR=NBET
:: GOSUB 1770 :: NBET=VAR
770 MOINS=0 :: PLUS=5 :: VAR=DEP :
: GOSUB 1770 :: DEP=VAR
780 MOINS=28 :: PLUS=40 :: VAR=LAR
G :: GOSUB 1770 :: LARG=VAR
790 IF NBET*LARG<=NCOL-NBET*DEP TH
EN 810
800 CALL NET :: CALL TROP :: CALL
ENT :: CALL STOP :: GOTO 720
810 MOINS=1 :: PLUS=20 :: VAR=NREP
S :: GOSUB 1770 :: NREPS=VAR
820 MOINS=0 :: PLUS=10 :: VAR=ESP
:: GOSUB 1770 :: ESP=VAR
830 GOSUB 1150 :: CALL NET
840 CALL IMP :: FOR RECNO=1 TO TOT
:: GOSUB 970 :: GOSUB 1680 ::
NEXT RECNO
850 IF ENR=1 THEN 890
860 FOR J=ENR TO NBET
870 FOR K=1 TO 3 :: ETIQ$(J,K)=""
:: NEXT K
880 NEXT J :: GOSUB 1320
890 PRINT #9:CHR$(27);"$" :: CLOSE
#9 :: GOTO 250
900 CLOSE #1 :: END
905 |-----
910 | s/p lect/enr fichier
915 |-----
920 OPEN #1:"DSK1.DONNEES",RELATIV

```

```

E,INTERNAL,FIXED 127
930 INPUT #1,REC 0:TRI$,TOT$,TITRE
$,IMP$,G$(1),G$(2),G$(3),G$(4)
,G$(5),G$(6),G$(7),G$(8),NG$
935 TRI=VAL(TRI$):: TOT=VAL(TOT$):
: NG=VAL(NG$):: RETURN
940 TRI$=STR$(TRI):: TOT$=STR$(TOT
):: NG$=STR$(NG)
945 PRINT #1,REC 0:TRI,TOT,TITRE$,
IMP$,G$(1),G$(2),G$(3),G$(4),G
$(5),G$(6),G$(7),G$(8),NG :: R
ETURN
950 | s/p saisie+tri enrég.
955 |-----
960 RECNO=VAL(PT$(Y))
970 INPUT #1,REC RECNO:D$(1),D$(2)
,D$(3),D$(4),D$(5),D$(6),D$(7)
,D$(8),D$(9)
980 CALL KEY(0,K,S):: IF NOT S THE
N 1070
990 IF K=15 THEN RECNO=TOT+1 :: RE
TURN
1000 IF K<>32 THEN 1070
1010 CALL SCREEN(11)
1020 CALL KEY(0,K,S)
1030 IF (S=1)*(K=32) THEN 1060
1040 CALL SOUND(-1,1000,12)
1050 GOTO 1020
1060 CALL SCREEN(7)
1070 M=1
1080 IF D$(1)="E" THEN RETURN
1090 IF INACTIF$="N" THEN 1110
1100 IF D$(2)="NNNNNNNN" THEN M=13
:: RETURN
1110 FOR M=1 TO NG
1120 IF SEG$(D$(2),M,1)=SEG$(GROUP
$,M,1) THEN M=12
1130 NEXT M :: RETURN
1135 |-----
1140 | Format de l'etiquette
1145 |-----
1150 CALL NET :: CALL METTRE :: CA
LL ENT :: CALL STOP
1160 RESTORE 1990 :: FOR I=1 TO 3
:: READ ETIQ$(1,I)
1170 ETIQ$(2,I)=ETIQ$(1,I)
1180 ETIQ$(3,I)=ETIQ$(1,I)
1190 ETIQ$(4,I)=ETIQ$(1,I)
1200 NEXT I
1210 ON NBET GOTO 1250,1240,1230,1
220
1220 GOSUB 1310
1230 GOSUB 1310
1240 GOSUB 1310
1250 GOSUB 1310
1260 CALL ALIGN
1270 ACCEPT AT(23,26)VALIDATE("ON"
)SIZE(-1):A$
1280 IF A$="N" THEN 1210
1290 RETURN
1295 |-----
1300 | Routine d'impression
1305 |-----
1310 IF NBET>ENR THEN ENR=ENR+1 ::
RETURN
1320 PRINT #9:CHR$(27);EP$ :: DEP$
=RPT$(" ",DEP)
1330 FOR I=1 TO 3
1340 FOR J=1 TO NBET
1350 PRINT #9:DEP$&SEG$(ETIQ$(J,I)
&BLANC$,1,LARG);
1360 NEXT J
1370 NEXT I
1380 PRINT #9
1390 ENR=1
1400 FOR I=1 TO ESP
1410 PRINT #9
1420 NEXT I
1430 RETURN
1435 |-----
1440 | Choix groupes a impr.
1445 |-----
1450 RESTORE 1970 :: CALL NET
1460 IF NG=0 THEN GROUP$="00000000
" :: GOTO 1570
1470 READ A$ :: DISPLAY AT(7,1):A$
: " -----"
1480 READ A$ :: GROUP$=""
1490 FOR I=1 TO 8
1500 IF (G$(I)="" )+(G$(I)="" ) THEN

```

```

1540
1510 DISPLAY AT(9+I,3)BEEP:G$(I);T
AB(15);A$
1520 ACCEPT AT(9+I,20)VALIDATE("ON
")SIZE(-1):GR$
1530 IF GR$="O" THEN 1550
1540 GR$="O"
1550 GROUP$=GROUP$&GR$
1560 NEXT I
1570 DISPLAY AT(10+I,3):"INACTIFS"
;TAB(15);"(O/N)N"
1580 ACCEPT AT(10+I,20)VALIDATE("O
N")SIZE(-1):INACTIFS
1590 IF INACTIFS="O" THEN RETURN
1600 IF GROUP$="00000000" THEN 250
ELSE RETURN
1605 |-----|
1610 | Fin d'impression
1615 |-----|
1620 RESTORE 2050 :: READ A$,B$
1630 PRINT #9:CHR$(10);CHR$(10)
1640 PRINT #9:" ";A$;CHR$(10);"
";B$
1650 PRINT #9:CHR$(10);CHR$(10)
1660 RETURN
1665 |-----|
1670 | Impression etiquette
1675 |-----|
1680 IF M<12 THEN RETURN
1690 FOR K=1 TO NREPS
1700 ETIQ$(ENR,1)=D$(8)&" "&D$(3)
1710 ETIQ$(ENR,2)=D$(4)
1720 ETIQ$(ENR,3)=D$(5)&" "&D$(6)
1730 GOSUB 1310
1740 NEXT K
1750 RETURN
1755 |-----|
1760 |Saisie options etiq.
1765 |-----|
1770 READ A$,T$,B$ :: DISPLAY AT(1
1,2):A$: :: TAB(2);VAR: :: T$
1780 ACCEPT AT(17,23)VALIDATE("ON"
)SIZE(-1)BEEP:A$
1790 IF A$="N" THEN 1830
1800 DISPLAY AT(23,1):B$
1810 ACCEPT AT(23,25)SIZE(-3)VALID
ATE(DIGIT)BEEP:VAR
1820 IF VAR<MOINS OR VAR>PLUS THEN
1810
1830 CALL NET2 :: RETURN
1835 |-----|
1840 | Fichier index
1845 |-----|
1850 OPEN #3:"DSK1.INDEX",INTERNAL
,INPUT,FIXED 4
1860 FOR I=1 TO TRI
1870 INPUT #3:P$(I)
1880 NEXT I
1890 CLOSE #3
1900 RETURN
1905 |-----|
1906 | Zone des datas
1907 |-----|
1910 DATA "NB DE COLONNES MAXIMUM"
," MODIFICATION (O/N)? N","
NOUVEAU NOMBRE(32-132)?"
1920 DATA "NB D'ETIQUETTES PAR LIG
NE"," MODIFICATIONS (O/N)? N"
," NOUVEAU NOMBRE (1-4) ? "
1930 DATA "MARGE DE GAUCHE"," MODI
FICATIONS (O/N)? N"," NOUVEAU
NOMBRE (0-5) ? "
1940 DATA "LARGEUR DE L'ETIQUETTE"
," MODIFICATIONS (O/N)? N","N
OUELLE LARGEUR(28-40)?"
1950 DATA "NB DE CHAQUE EXEMPLAIRE
"," MODIFICATIONS (O/N)? N","
NOUVEAU NOMBRE (1-20)?"
1960 DATA "ECCART ENTRE ETIQUETTES
"," MODIFICATIONS (O/N)? N","
NOUVEAU NOMBRE (0-10)?"
1970 DATA " GROUPES A IMPRIMER"
1980 DATA (O/N)O
1990 DATA "TITRE..NOM...PRENOM....
....."
2000 DATA "ADRESSE....."
....."
2010 DATA "CODE.POSTAL....VILLE...
....."
2020 DATA " NOUVEAU DISPOSITIF ? "

```

```

2030 DATA "NOM PRENOM",ADRESS
E,CODE VILLE," TEL","
DIVERS"
2040 DATA -----,
-----,
-----,
-----,
-----,
2050 DATA -----,FIN D'I
MPRESSION
2055 |-----|
2056 | s/programmes fermes
2057 |-----|
2060 SUB TITRE :: DISPLAY AT(9,10)
:"TI 99/4A": :: TAB(4);"IMPRESS
ION DU FICHIER":TAB(4);"-----
-----" :: SUBEND
2070 SUB ENT :: DISPLAY AT(23,6):"
APPUYEZ SUR ENTER" :: SUBEND
2080 SUB RIEN :: DISPLAY AT(8,2):"
PAS DE FICHIER SUR DISQUE": ::
TAB(9);"OPTIONS": :: "
1. NOUVEL ESSAI": :: 2. S
ORTIE" :: SUBEND
2090 SUB OPT :: DISPLAY AT(6,4):"O
PTIONS D'IMPRESSION":TAB(4);"
-----"
2100 BISPLAY AT(10,3):"1. TOUT LE
FICHIER": :: TAB(3);"2. FICHIER
ORGANISE": :: TAB(3);"3. ETIQU
ETTES": :: TAB(3);"4. SORTIE" ::
SUBEND
2110 SUB TROP :: DISPLAY AT(11,6):
"LIGNE D'IMPRESSION": :: TAB(9)
;"TROP LONGUE" :: SUBEND
2120 SUB METTRE :: DISPLAY AT(8,5)
:"METTEZ LES ETIQUETTES": :: "
SUR L'IMPRIMANTE" :: SUB
END
2130 SUB ALIGN :: DISPLAY AT(23,2)
:"ALIGNEMENT CORRECT(O/N)?O"
:: SUBEND
2140 SUB DISP(IMP$):: DISPLAY AT(8
,1):" DISPOSITIF D'IMPRESSION
": :: TAB(3);IMP$
2150 DISPLAY AT(14,1)BEEP:" MODIFI
CATIONS (O/N)?N" :: SUBEND
2160 SUB IMP :: DISPLAY AT(12,1):"
L'IMPRESSION EST EN COURS":"
-----" ::
SUBEND
2170 SUB NET :: CALL VCHAR(1,3,32,
672):: SUBEND
2180 SUB STOP
2190 CALL SOUND(150,800,8)
2200 CALL KEY(0,K,S):: IF NOT S TH
EN 2200 ELSE IF K<>13 THEN 21
90
2210 SUBEND
2220 SUB NET2 :: FOR I=11 TO 23 ST
EP 3 :: DISPLAY AT(I,1):: NEX
T I :: SUBEND

```

```

5 |-----|
6 | PROGRAMME DE RECHERCHE
7 |-----|
10 ON BREAK NEXT
20 CALL CLEAR
30 DIM P$(450),D$(4)
40 CALL SCREEN(7):: CALL COLOR(13,
11,1)
50 CALL VCHAR(1,31,129,96)
60 FOR A=1 TO 12 :: CALL COLOR(A,5
,12):: NEXT A
70 CALL TITRE :: CALL ENT :: CALL
STOP(K)
80 DISPLAY AT(23,6):"UN INSTANT SV
P..."
90 OPEN #1:"DSK1.",RELATIVE,INTERN
AL,INPUT
100 INPUT #1:T$
110 IF T$="DONNEES" THEN 160 ELSE
IF T$<>" " THEN 100
120 CLOSE #1
130 CALL NET :: CALL RIEN :: CALL
CHOIX
140 ACCEPT AT(22,22)SIZE(-1)VALIDA
TE("12")BEEP:I
150 IF I=2 THEN 390 ELSE 80
160 CLOSE #1
170 OPEN #1:"DSK1.DONNEES",RELATIV
E,INTERNAL,FIXED 127
180 INPUT #1:TRI$,TOT$ :: TRI=VAL(
TRI$):: TOT=VAL(TOT$)
190 OPEN #3:"DSK1.INDEX",INPUT,IN
TERNAL,FIXED 4 :: FOR I=1 TO T
RI
200 INPUT #3:P$(I)
210 NEXT I :: CLOSE #3
220 IF TOT=0 THEN 380
230 CALL NET :: CALL RECH
240 ACCEPT AT(24,2)VALIDATE(UALPHA
):T$: :: IF T$=" " THEN 290
250 L=LEN(T$)
260 IF L<6 THEN 450
270 CALL NET :: CALL TROP :: CALL
ENT :: CALL STOP(K):: GOTO 230
290 CALL NET
300 FOR I=1 TO TOT :: IF I<=TRI TH
EN Z=VAL(P$(I))ELSE Z=I
310 GOSUB 400
320 PRINT USING "###":I:: PRINT "
"&D$(3)
330 NEXT I
340 CALL HCHAR(12,1,42,2)
350 CALL HCHAR(12,31,42,2)
360 CALL FIN :: CALL STOP(K)
370 CALL VCHAR(1,31,129,96):: IF K
=15 THEN 230
380 CLOSE #1
390 END
395 |-----|
400 INPUT #1,REC Z:D$(1),D$(2),D$(
3):: D$(4)=SEG$(D$(3),1,L)
480 IF D$(4)=T$ THEN 530
490 IF D$(4)<T$ THEN I=ZT+1 ELSE J
=ZT-1
500 IF (ZT<1)+(I>J)=0 THEN 460 ELS
E 570
510 INPUT #1,REC Z:D$(1),D$(2),D$(
3):: D$(4)=SEG$(D$(3),1,L)
520 IF D$(4)<>T$ THEN 540
530 IF ZT>1 THEN ZT=ZT-1 :: Z=VAL(
P$(ZT)): GOTO 510
540 INPUT #1,REC Z:D$(1),D$(2),D$(
3):: D$(4)=SEG$(D$(3),1,L)
550 IF D$(4)=T$ THEN 560 ELSE ZT=Z
T+1 :: Z=VAL(P$(ZT)): GOTO 54

```

## Recherche sur un fichier

Vous pouvez afficher la totalité des noms en appuyant seulement sur <enter> à la question <début du nom à rechercher> sinon, le programme affichera tous les noms commençant par la (ou les) lettre(s) choisie(s) (6 maximum). Ce programme, comme les autres, tourne plus rapidement si le fichier est classé alphabétiquement par index.

### Programme "RECHERCHER"

Basic étendu

```

0
560 IF D$(4)=T$ THEN PRINT USING "
### ":ZT;: PRINT D$(3)ELSE 57
0
565 IF ZT<>TRI THEN ZT=ZT+1 :: Z=V
AL(P$(ZT)):: INPUT #1,REC Z:D$(
1),D$(2),D$(3):: D$(4)=SEG$(D
$(3),1,L):: GOTO 560
570 IF TRI=TOT THEN 610
580 FOR Z=TRI+1 TO TOT :: INPUT #1
,REC Z:D$(1);D$(2);D$(3)
590 D$(4)=SEG$(D$(3),1,L):: IF D$(
4)=T$ THEN PRINT USING "### ":
Z;: PRINT D$(3)
600 NEXT Z
610 CALL FIN :: CALL STOP(K):: IF
K=15 THEN 230 ELSE STOP
620 SUB TITRE :: DISPLAY AT(10,10)
:"TI 99/4A": :TAB(4);"RECHERCH
E SUR FICHIER":TAB(4);"~~~~~
~~~~~" :: SUBEND
630 SUB ENT :: DISPLAY AT(23,6):"A
PUYEZ SUR ENTER" :: SUBEND
635 SUB RIEN :: DISPLAY AT(8,6):"A
BSENCE DE FICHIER":TAB(6);"~~~~
~~~~~": : :TAB(8);"O
PTIONS"
640 DISPLAY AT(14,5):"1. NOUVEL ES
SAI": :TAB(5);"2. SORTIE" :: S
UBEND
650 SUB CHOIX :: DISPLAY AT(22,8):
"VOTRE CHOIX ?" :: SUBEND
660 SUB RECH :: DISPLAY AT(22,1)BE
EP:" DEBUT DU NOM A RECHERCHER
" :: SUBEND
670 SUB TROP :: DISPLAY AT(12,1)BE
EP:" IL Y A TROP DE LETTRES !!
": : (5 MAXIMUM)" :: SUB
END
680 SUB NET :: CALL VCHAR(1,3,32,6
72):: SUBEND
690 SUB STOP(K)
700 CALL SOUND(150,800,2)
710 CALL KEY(0,K,S):: IF NOT S THE
N 710
720 IF K<>15 AND K<>13 THEN 700
730 SUBEND
740 SUB FIN :: PRINT ":" <BACK>ou<E
NTER>pour arret" :: SUBEND

```

## Titrer en gros

Ce programme dépend directement des possibilités de votre imprimante. A cet effet, il comporte une bibliothèque de toutes les spécifications employées dans les différents programmes afin que vous puissiez faire une adaptation. Le titre est en double largeur, caractères italiques, impression en gras. Une fois terminé, le sommaire est rechargé automatiquement.

### Programme "TITRER"

#### Basic étendu

```

5 |*****
6 |* TITRER EN GROS
7 |*****
10 CALL CLEAR :: CALL SCREEN(7)::
CALL COLOR(13,7,7)
20 CALL VCHAR(1,31,129,96)
30 FOR I=1 TO 12 :: CALL COLOR(I,5
,12):: NEXT I
40 DISPLAY AT(8,10):"TI 99/4A": :
:" TITRER EN GROS CARACTERES":
"~~~~~
~~~~~"
45 DISPLAY AT(17,3):"<FCTN AID>:sp
ecifications": : "<ENTER>:c

```

```

ontinuer"
50 CALL SOUND(150,800,5)
60 CALL KEY(0,K,S):: IF NOT S THEN
60 ELSE IF K=13 THEN 120 ELSE
IF K<>1 THEN 60
65 CALL NET :: DISPLAY AT(1,2):"IM
PRIMANTE SEIKOSHA 550 A": "~~~~
~~~~~"
70 DISPLAY AT(4,1):"CHR$(14) carac
tere dilate": "CHR$(15) fin de d
ilatation": "CHR$(24) vide le bu
ffer"
80 DISPLAY AT(7,1):"CHR$(16);""XXX
"" tabulation": "CHR$(10) retour
chariot": : : "CHR$(27);annonce
les donnees": ""C"" condenses
(136/ligne)"
90 DISPLAY AT(13,1):""B"" caracte
res italique": ""#"" caracteres
gras": ""$"" fin de caracteres
gras": ""N"" caracteres normaux
": : "IMP$&""CR""?"
100 DISPLAY AT(20,1):""CR"" supp
rime le retour du chariot au 8
0 lème caractère(pour en mettr
e jusqu'a 136)": : "<enter> p
our continuer"
110 CALL KEY(0,K,S):: IF NOT S THE
N 110
120 CALL NET :: OPEN #9:"PIO"
130 DISPLAY AT(8,2)BEEP:"TITRE EN
GROS CARACTERES": "~~~~~
~~~~~": : : "Donnee: 1
ligne maximum"
140 ACCEPT AT(16,1)SIZE(-28):A$ ::
IF A$="" THEN 210
150 DISPLAY AT(20,4):"IMPRESSION E
N COURS":TAB(4);"~~~~~
~~~~~"
160 PRINT #9:CHR$(27);"B";CHR$(14)
170 X=INT((40-LEN(A$))/2)
180 PRINT #9:CHR$(27);"#";TAB(X);A
$
190 PRINT #9:TAB(X-2);RPT$("~",LEN
(A$))
200 PRINT #9:CHR$(15);CHR$(27);"$"
;CHR$(27);"N";: :
210 CLOSE #9
220 RUN "DSK1.LOAD"
230 SUB NET :: CALL VCHAR(1,3,32,6
72):: SUBEND

```

# Programmes en assembleur

## Saisie

Lorsque vous entrerez ces programmes, NE SAISISSEZ PAS LES COMMENTAIRES, car vous ne pourriez pas les assembler sur une seule face de disquette. Si vous disposez de plusieurs lecteurs, aucun problème ; il suffira d'envoyer le fichier objet sur une autre disquette. Le programme est prêt à être transformé en "image mémoire" grâce au programme "SAVE" de la disquette Editeur/Assembleur Texas Instruments (une fois assemblé), il se chargera alors environ trois fois plus vite. Si vous l'utilisez avec la Mini-mémoire, qui ne permet pas de charger ce type de fichier, rajoutez "FICHES" après le END du fichier source "FICH0" pour que votre programme démarre automatiquement.

## Initialiser le fichier

C'est le premier programme à faire tourner. Il permet d'ouvrir les fichiers et de contrôler si l'espace initial pour les données est suffisant. Vous devrez alors

rentrer le nom des groupes dans lesquels vous mettrez vos fiches (au moins une) et indiquer le nom de votre imprimante (ce n'est pas obligatoire). Vous pourrez modifier ce nom et les paramètres éventuels à chaque fois que vous choisirez l'option "IMPRESSION".

## Gestion du fichier

C'est le programme qui permet de saisir les données et, éventuellement, de modifier les noms de groupes ou les informations contenues dans les fiches. Il permet également de marquer une fiche pour l'effacer (avec "TRI"), de rechercher une fiche ou un groupe et de l'afficher sur l'écran. Chaque enregistrement contient les informations relatives à l'adresse, plus un caractère : soit A (actif), I (inactif) ou E (effacement), précédant les huit octets (groupe dans lequel est la personne). Evitez de modifier les saisies concernant les noms, prénoms, adresses, codes, villes et numéros de téléphone. En revanche, les autres informations sont spécifiques et vous pourrez les changer à votre convenance.

## Source "FICH0"

```

COPY "DSK1.FICH1"
COPY "DSK1.FICH2"
COPY "DSK1.FICH3"
COPY "DSK1.FICH4"
SLAST END ***** rajouter ici FICHES
* si vous ne voulez utiliser que le
* fichier classique avec départ automatique

```

Pour effectuer des modifications, il faut, dans "PROGRAMME DE GESTION" et à l'étiquette "TEXT6", changer l'intitulé des saisies. Dans "INITIALISER LE FICHIER", à l'étiquette "NBSAIS", il faut éventuellement les nouveaux chiffres, chaque chiffre indiquant le nombre maximum de chaque



rubrique (nom, adresse, code, ville, téléphone, etc...) dans l'ordre ou vous les avez mis dans "TEXT6", le total ne devant pas dépasser 127. Dans "IMPRIMER LE FICHIER", à l'étiquette "ENTET1" et "ENTET2", modifier les intitulés. Dans le même fichier, mais cette fois à l'étiquette "IMPR5" vous trouverez les paramètres d'impression :

- TLIGNE correspond à l'imprimante ;
- TPAB correspond à l'emplacement où se trouve l'information sur l'enregistrement.

## Trier le fichier

Dès que vous avez une dizaine de fiches, classez-les par ordre alphabétique. Ce programme va créer un index qui permettra aux autres programmes d'aller chercher les informations alphabétiquement et vous pourrez, en option, effacer des fiches. Utilisez-le fréquemment ; il est très rapide. Vous pourrez également trier les autres informations, sauf les adresses et les chiffres d'inégales longueur, car le programme ne prend en compte que quatre octets au maximum.

## Imprimer le fichier

Vous pouvez organiser un fichier organisé sous forme de colonnes. Le choix des groupes à imprimer est aussi possible. Le programme utilise une imprimante avec caractères condensés (136 caractères par lignes) ; cela permet l'affichage de la totalité des informations sur une ligne. Dans le cas d'une imprimante sans ce type de caractère, il faudra modifier, dans "IMPR5" comme indiqué plus haut, les saisies dans TPAB, leur report dans "TLIGNE", et se restreindre à un nombre de caractères plus limité.

Vous pouvez imprimer des étiquettes d'adresses ; le programme est conçu pour une imprimante 80 colonnes pouvant imprimer deux étiquettes de front.

De nombreux paramètres vous permettront de centrer vos étiquettes aux mieux. J'utilise pour ma part deux étiquettes de front (mais vous pouvez n'en utiliser qu'une) et les paramètres que j'utilise sont stockés dans le fichier "ETIQUETTES D'ADRESSES" aux étiquettes MG (marge de gauche), ENTRE (écart entre les deux étiquettes), SAUT (nombre de lignes à sauter entre les séries), et enfin MAXI (nombre maximum de lignes de l'imprimante). Il y a également, à l'étiquette HEX10, des codes LF "CHR\$(10)" commun à toutes les imprimantes, et qui provoquent l'impression de la mémoire tampon, un retour chariot et un saut de ligne.

Une étiquette modèle est imprimée pour déterminer le format maximum de vos données. Toutes les spécifications de l'imprimante qui sont employées sont dans "IMPRESSION DU FICHIER", aux étiquettes COMPR (caractères compressés), BIGCAR (gros caractères + caractères gras) et FINBIG (fin gros caractères, fin caractères gras et remise en mode normal).

A vous de placer les codes correspondants pour votre imprimante. Dans le cas où vous n'auriez pas toutes ces possibilités, il faudra mettre des >00 à la place des codes manquants ; de cette façon, vous n'aurez rien d'autre à modifier.

99

## Source "FICH1"

```

DEF FICHES,SFIRST,SLOAD,SLAST
REF VSBW,VMBW,VMBR,KSCAN
REF DSRLNK,VWTR,XMLLNK,VSBR
AORG >A000

SFIRST
SLOAD B @FICHES
TEXTE TEXT -'XXXXXXXXXXXXXXXXXXXXXXX '
TEXT -'X FICHIER D'ADRESSES X '
TEXT -'XXXXXXXXXXXXXXXXXXXXXXX '
TEXT -'SOMMAIRE '
TEXT -'~~~~~ '
TEXT -'***** '
TEXT -'* * '
TEXT -'* Appuyez * '
TEXT -'* sur * '
TEXT -'* 1 pour Gerer * '
TEXT -'* 2 Imprimer * '
TEXT -'* 3 Initialiser * '
TEXT -'* 4 Trier * '
TEXT -'* 5 Terminer * '
TEXT -'* * '
TEXT -'***** '

ADDRE DATA 48,88,128,255,295,406,446
DATA 486,526,566,606,646,686,726,766
DATA 806,0
REG EQU >20BA
TPAB EQU >1000
PAB EQU >F80
PAB2 EQU >F00
RTCLA EQU >8375
PNTR EQU >8356
* mode input int/fix 38
DSK0 DATA >000D,>1000,>2626,>0000,>0005
TEXT 'DSK1.'
* mode input int/fix 127
DSK1 DATA >000D,>1000,>7F7F,>0000,>000C
TEXT 'DSK1.DONNEES'
* mode input int/var 254
DSK2 DATA >000C,>1000,>FEFE,>0000,>000A
TEXT 'DSK1.INDEX'
* emplacement des noms de groupes
GROUPE DATA >2020,>2020,>2020,>2020,>2020
DATA >2020,>2020,>2020
DATA >2020,>2020,>2020,>2020,>2020
DATA >2020,>2020,>2020
DATA >2020,>2020,>2020,>2020,>2020
DATA >2020,>2020,>2020
DATA >2020,>2020,>2020,>2020,>2020
DATA >2020,>2020,>2020
DATA >2020,>2020,>2020,>2020
TRI DATA 0
TOT DATA 0
NGR DATA 0
A BYTE 65
E BYTE 69
I BYTE 73
ZERO BYTE 48
UN BYTE 49
DEUX BYTE 50
TROIS BYTE 51

```

```

QUATRE BYTE 52
CINQ  BYTE 53
NEUF   BYTE 57
ENT    BYTE 13
OUI    BYTE 79
NON    BYTE 78
BACK   BYTE 15
REDO   BYTE 6
FCTNS  BYTE 8
FCTND  BYTE 9
INDX   DATA 0
TERM   DATA 0
DRAP   DATA 0
DRAP1  DATA 0
DRAP2  DATA 0
D450   DATA 450
DIX    DATA 10
SECT   DATA 0
LGR    DATA 0
NLIG   DATA 0
LSAIS  DATA 0
D40    DATA 40
MINI   DATA 227
D320   DATA 320
D11    DATA 11
D33    DATA 33
D80    DATA 80
ATTRIB DATA >0100
TAMPON BSS 118
BUFGR  BYTE 0
      BSS 8
      EVEN

* s/p initialisation "tampon"
INITAM CLR R2
TA      MOV B R1, @TAMPON(R2)
      INC R2
      CI R2, 118
      JNE TA
      RT

* s/p efface ecran
CLEAR1 LI R0, 160
      JMP CL1
CLEAR  CLR R0
CL1    LI R1, >2000
CL     BLWP @VSBW
      INC R0
      CI R0, 960
      JNE CL
      RT

* s/p affichage
AFFICH MOV *R3+, R0
      JEQ RET
AFF     MOV B *R2+, R1
      JLT AFFICH
      BLWP @VSBW
      INC R0
      JMP AFF
RET     RT

* s/p fermeture fichier
OVER   LI R0, PAB
      LI R1, ATTRIB
      LI R2, 2
      BLWP @VMBW
      LI R9, PAB+9

      MOV R9, @PNTR
      BLWP @DSRLNK
      DATA 8
      RT
      * s/p d'ouverture
OPEN   LI R0, PAB
      BLWP @VMBW
DSR    LI R9, PAB+9
      MOV R9, @PNTR
      BLWP @DSRLNK
      DATA 8
      JNE RET2
      MOV @INDX, @INDX
      JNE RET3
      B @ERREUR
RET2    RT
RET3    SETO @TERM
      RT
      * s/p d'arret o/n
ARRET  BLWP @KSCAN
      MOV B @>837C, R4
      JEQ ARRET
      MOV B @RTCLA, R4
      RT
      * s/p de saisie
ACCEPT LI R2, 1
SG1     LI R3, >100
SG2     BLWP @KSCAN
      MOV B @>837C, R4
      JEQ SG16
      CB @RTCLA, @ENT
      JEQ SG12
      CB @RTCLA, @BACK
      JNE RED
      MOV @DRAP6, @DRAP6
      JEQ SG15
      MOV R12, R12
      JNE SG17
      B @NOMGR
SG17    B @NOUVE
SG15    MOV @DRAP1, @DRAP1
      JEQ SG13
      B @OPT
SG13    MOV @DRAP, @DRAP
      JEQ SG2
      B @IMPR
SG12    B @FINI
SG16    B @CURS
RED     CB @RTCLA, @REDO
      JNE SG7
      MOV B *R7, R1
      BLWP @VSBW
      MOV @DRAP4, @DRAP4
      JNE SG2
      MOV @DRAP, @DRAP
      JEQ SG5
      B @SAISIE
SG5     MOV @DRAP2, @DRAP2
      JEQ SG6
      B @AJGR2
SG6     MOV @DRAP1, @DRAP1
      JEQ SG7
      B @AJOUT
SG7     CB @RTCLA, @FCTNS

      JNE SG3
      CI R2, 1
      JEQ SG2
      MOV B *R7, R1
      BLWP @VSBW
      DEC R2
      DEC R7
      DEC R0
      JMP SG2
SG3     CB @RTCLA, @FCTND
      JNE SG4
      MOV @DRAP2, @DRAP2
      JNE SG2
      C R2, @LSAIS
      JEQ SG2
      MOV B *R7, R1
      BLWP @VSBW
      C R2, @LSAIS
      JHE SG11
      INC R0
      INC R2
      INC R7
      JMP SG2
SG4     MOV B @RTCLA, R1
      MOV R12, R12
      JNE SG18
      MOV @DRAP2, @DRAP2
      JEQ SG8
SG18    MOV @DRAP6, @DRAP6
      JNE SG10
      CB R1, @OUI
      JEQ SG8
      CB R1, @NON
      JNE SG2
      JMP SG8
SG10    CB R1, @ZERO
      JL SG2
      CB R1, @NEUF
      JH SG2
SG8     MOV B R1, *R7+
      BLWP @VSBW
      MOV @DRAP2, @DRAP2
      JNE FINI1
      C R2, @LSAIS
      JHE SG11
      INC R0
      INC R2
      JMP SG14
SG11    DEC R7
      JMP SG14
CURS    DEC R3
      JNE SG14
      MOV R6, R6
      JEQ SGCL
      CLR R6
      LI R1, >1E00
      BLWP @VSBW
      JMP SG9
SGCL    SETO R6
      MOV B *R7, R1
      BLWP @VSBW
      JMP SG9
FINI1   DEC R7
      * apres enter

```



```

TEXT -'VILLE: '
TEXT -'TELEPHONE: '
TEXT -'COTISATION: '
TEXT -'No LIC: '
TEXT -'BOX: '
ADD6 DATA 187,205,280,363,442
DATA 525,601,680,764,847,0
TEXT7 TEXT 'Valider : <ENTER> Corriger : '
TEXT -'<BACK> '
ADD7 DATA 922,0
TEXT8 TEXT -'Une autre fiche ? O/N '
ADD8 DATA 927,0

TEXT9 TEXT 'indiquez par O/N le groupe dans '
TEXT -'lequel '
TEXT 'vous voulez mettre le nouveau '
TEXT -'nom '
ADD9 DATA 280,322,0
TEXT10 TEXT 'LES 450 FICHES PREVUES SONT '
TEXT -'UTILISEES '
TEXT -'Initialisez un autre fichier '
TEXT -'sur une autre disquette '
TEXT -'Appuyez sur une touche '
ADD10 DATA 401,524,566,847,0
EVEN

```

#### \* s/p evite repetition touche

```

BLOQ MOV B @>837C,R13
ANDI R13,>FF00
LI R14,>2000
COC R14,R13
RT

```

#### \* s/p d'écriture

```

WRITE LI R0,TPAB
LI R1,GROUPE
LI R2,94
BLWP @VMBW
AI R0,94
LI R1,NOMIMP
LI R2,33
BLWP @VMBW
RT

```

\*\*\*\*\*

#### \* ouvre le fichier "index"

```

GERER MOV @INDX2,@INDX2
JNE DON1
LI R1,DSK2
BL @SATTR
LI R2,20
BL @OPEN
SETO @INDX
CLR R5

```

#### \* lecture des enregistrements

```

INDEX1 BL @INPUT
BL @DSR
MOV @TERM,@TERM
JEQ TRANSF

```

#### \* verifier si err=5 (fin fichier)

```

CLR @TERM
LI R0,PAB+1
BLWP @VSBW
SRL R1,13
CI R1,5
JEQ FERME
B @ERREUR

```

#### \* fermeture "index"

```

FERME BL @OVER
CLR @INDX
SETO @INDX2
JMP DON1

```

#### \* transfer dans "index"

```

TRANSF LI R0,TPAB
LI R1,INDEX
A R5,R1
LI R2,254
BLWP @VMBW

```

```
AI R5,254
```

```
JMP INDEX1
```

#### \* ouverture fichier "donnees"

```

DON1 MOV @DRAP10,@DRAP10
JEQ DON2
B @IMPR1

```

```

DON2 LI R1,DSK1
BL @SATTR
LI R2,22
BL @OPEN

```

#### \* lecture enregistrement 0

```
BL @INPUT
BL @DSR

```

#### \* saisie groupes,tri,tot,ng,imp

```
BL @READ
```

#### \* ecriture a l'ecran

```

ECRAN1 BL @CLEAR1
LI R2,TEXT1
LI R3,ADD1
BL @AFFICH
LI R2,394
LI R7,391
MOV @TRI,R4
BL @DECI
AI R2,80
AI R7,80
MOV @TOT,R4
BL @DECI
AI R2,80
AI R7,80
MOV @NGR,R4
BL @DECI
LI R0,708
LI R1,NOMIMP+1
LI R2,32
BLWP @VMBW
BL @ENTER

```

#### \* ecran suivant

```

OPT BL @CLEAR
LI R2,TEXT2
LI R3,ADD2
BL @AFFICH
CLR R12
CLR @DRAP
CLR @DRAP1
CLR @DRAP2
CLR @DRAP3
CLR @DRAP4
CLR @DRAP5
CLR @DRAP6

```

```
CLR @DRAP7
```

```
CLR @DRAP8
```

```
CLR @DRAP9
```

```
CLR @DRAP10
```

```
CLR @DRAP11
```

#### \* choix

```

CHOIX1 BLWP @ARRET
BL @BLOQ
JNE CHOIX1
CB R4,@UN
JNE CH7
JMP JOUR
CH7 CB R4,@DEUX
JNE CH8
B @NOMGR
CH8 CB R4,@TROIS
JNE CH9
B @RECHER
CH9 CB R4,@QUATRE
JNE CHOIX1

```

#### \* ferme le fichier

```
BL @OVER
B @FICH
```

#### \* options mise a jour

```

JOUR BL @CLEAR
LI R2,TEXT3
LI R3,ADD3
BL @AFFICH
CHOIX2 BLWP @ARRET
BL @BLOQ
JNE CHOIX2
CB R4,@UN
JNE CH10
B @AJOUTE
CH10 CB R4,@DEUX
JNE CH11
B @MODIF
CH11 CB R4,@TROIS
JNE CH12
B @EFFACE
CH12 CB R4,@QUATRE
JNE CHOIX2
B @OPT

```

#### \* options groupes

```

NOMGR BL @CLEAR
CLR @DRAP6
LI R2,TEXT4
LI R3,AD4
BL @AFFICH
CHOIX3 BLWP @ARRET

```



```

BL @BLOQ
JNE CHOIX3
CB R4,@UN
JL CHOIX3
CB R4,@TROIS
JH CH14
B @MODGR
CH14 CB R4,@QUATRE
JNE CH18
B @APPGR
CH18 CB R4,@CINQ
JNE CHOIX3
B @OPT
* options de recherche
RECHER BL @CLEAR
LI R2,TEXT5
LI R3,ADD5
BL @AFFICH
CHOIX4 BLWP @ARRET
BL @BLOQ
JNE CHOIX4
CB R4,@UN
JNE CH15
B @RECH1
CH15 CB R4,@DEUX
JNE CH16
B @AFFIGR
CH16 CB R4,@TROIS
JNE CH17
B @TOUTFI
CH17 CB R4,@QUATRE
JNE CHOIX4
B @OPT
* s/p efface la dernière ligne
CLS24 LI R1,NET
LI R0,920
LI R2,40
BLWP @VMBW
RT

```

#### \* AJOUTER UN NOM

```

AJOUTE BL @CLEAR1
* test si tot<=450
C @TOT,@D450
JLE AJOUT2
LI R2,TEXT10
LI R3,ADD10
BL @AFFICH
BL @OVER
BL @ARRET
BL @FICH

```

#### \* test positif

```

AJOUT2 LI R1,>2E00
BL @INITAM
LI R2,TEXT6
LI R3,ADD6
BL @AFFICH
MOV @TOT,R4
INC R4

```

#### \* No de fiche

```

AJOUT4 LI R2,198
LI R7,195
BL @DECI

```

```

AJOUT SETO @DRAP1
CLR @DRAP
CLR @NLIG
CLR @LGR
LI R8,NBSAIS
* saisie des informations
AJOUT1 LI R0,212
A @NLIG,R0
CLR @DRAP2
MOV R0,R13 --> adr ecran
MOV *R8+,@LSAIS depart
JEQ AJFIN
LI R1,TAMPON
A @LGR,R1
MOV R1,R7
MOV @LSAIS,R2
BLWP @VMBW
BL @ACCEPT

```

#### \* verifie si appel depuis "modif"

```

MOV @DRAP4,@DRAP4
JNE AJOUT3

```

#### \* recherche dans fichier du nom

```

CI R13,292
JNE AJOUT3
MOV @TOT,@TOT
JEQ AJOUT3
LI R0,TPAB+4
LI R1,TAMPON+4
LI R2,28
BLWP @VMBW
BL @ADDIT
MOV @ASCII,@SOMME1
SETO @DRAP3
BL @RBLOC
MOV @DRAP3,@DRAP3
JNE AJOUT3
LI R8,NBSAIS+2
BL @CLS24
JMP AJOUT1

```

#### \* lignes suivantes

```

AJOUT3 A @D80,@NLIG
* LGR=nb a ajouter a "tampon"
A @LSAIS,@LGR
JMP AJOUT1

```

#### \* affiche <enter><back>

```

AJFIN LI R2,TEXT7
LI R3,ADD7
BL @AFFICH
CLR @DRAP3
BL @ENTER
MOV @DRAP2,@DRAP2
JNE AJFIN1
BL @CLS24
B @AJOUT

```

#### \* poursuite de l'operation

```

AJFIN1 MOV @DRAP4,@DRAP4
JNE AJFIN2
INC @TOT
AJFIN2 BL @CLEAR1
CLR @LSAIS
INC @LSAIS

```

#### \* affiche le nom

```

LI R0,165
LI R3,TAMPON

```

```

LI R2,32
AJG1 MOV *R3+,R1
CB R1,@POINT
JEQ AJG2
BLWP @VSBW
AJG2 INC R0
DEC R2
JNE AJG1

```

#### \* appartenance a un groupe

```

LI R2,TEXT9
LI R3,ADD9
BL @AFFICH
LI R2,TEXTG1
LI R3,ADGR+4
BL @AFFICH
AJGR MOV @DRAP4,@DRAP4
JNE AJGR2
LI R2,TEXTG3
LI R3,ADG3
BL @AFFICH
LI R3,9
LI R2,BUFGR
LI R1,>2000

```

```

BUFCL MOV *R2+,R1
DEC R3
JNE BUFCL

```

```

AJGR2 CLR @DRAP1
CLR @NLIG
CLR @LGR

```

#### \* affichage du groupe

```

AJGR1 LI R0,486
A @NLIG,R0
LI R1,GROUPE
A @LGR,R1
LI R2,11
BLWP @VMBW
INC @DRAP1

```

#### \* saisie O ou N

```

ON LI R0,498
A @NLIG,R0
LI R1,BUFGR+1
A @DRAP1,R1
DEC R1
BLWP @VSBW
MOV R1,R7
BL @ACCEPT

```

#### \* groupe suivant

```

C @DRAP1,@NGR
JEQ ENTBAC
A @D40,@NLIG
A @D11,@LGR
JMP AJGR1

```

```

ENTBAC CLR @DRAP1

```

#### \* enter ou back

```

LI R2,TEXT7
LI R3,ADD7
BL @AFFICH
CLR @DRAP2
BL @ENTER
MOV @DRAP2,@DRAP2
JNE GRTRAN
BL @CLS24
JMP AJGR2

```

#### \* transfer et verif. si inactif

```

GRTRAN CLR R3
      LI R5,118
      CLR @DRAP1
      LI R1,BUFGR
      LI R2,TAMPON+119
GRTR INC R1
      INC R3
      C R3,@NGR
      JH AJSUIT
      MOVB *R1,*R2+
      CB *R1,@OUI
      JNE GRTR
      SETO @DRAP1
      JMP GRTR
AJSUIT MOV @DRAP1,@DRAP1
      JEQ INACT
* marque actif
      MOVB @A,@TAMPON(R5)
      JMP AJENR
* marque inactif
INACT MOVB @I,@TAMPON(R5)
AJENR CLR @DRAP1
* transfert dans tpab
      LI R0,TPAB
      LI R1,TAMPON
      LI R2,127
      BLWP @VMBW
* No de l'enregistrement
AJENR4 LI R0,PAB+6
      MOV @DRAP4,@DRAP4

      JEQ AJENR1
      LI R1,NUMENR
      JMP AJENR2
AJENR1 LI R1,TOT
AJENR2 LI R2,2
      BLWP @VMBW
* mode update
      LI R0,PAB+1
      LI R1,>0900
      BLWP @VSBW
* enregistrement de la fiche
      BL @PRINT
      BL @DSR
      MOV @DRAP5,@DRAP5
      JNE AJ2
      MOV @DRAP4,@DRAP4
      JEQ AJENR3
      B @AUTMOD
* autre fiche ?
AJENR3 BL @CLS24
      LI R2,TEXT8
      LI R3,ADD8
      BL @AFFICH
ARR BL @ARRET
      CB R4,@OUI
      JEQ AJ1
      CB R4,@NON
      JNE ARR
      CLR @DRAP2
      JMP ENREGO

AJ1 B @AJOUTE
AJ2 B @AUTEFF
* enregistrer tri,tot,ngr...
ENREGO CLR @NUMENR
      BL @NUM
      BL @WRITE
      LI R0,PAB+1
      LI R1,>0900
      BLWP @VSBW
      BL @PRINT
      BL @DSR
      B @OPT
*****
* Dans chaque enregistrement
* ~~~~~
* ENR+0 = titre (4 octets)
* ENR+4 = nom prenom(28 octets)
* ENR+32 = adresse (28 octets)
* ENR+60 = code ptt (5 octets)
* ENR+65 = ville (20 octets)
* ENR+85 = telephone (15 octets)
* ENR+100= cotisation (4 octets)
* ENR+104= no de lic (6 octets)
* ENR+110= box (8 octets)
* ENR+118= act,ina,eff(1 octet)
* ENR+119= app.groupe (8 octets)
*
* TOTAL: 127 octets
*****

```

## Source "FICH2"

### \* PROGRAMME D'INITIALISATION

```

*****
TITRE4 TEXT -'INITIALISER LE FICHIER '
      TEXT -'~~~~~'
      TEXT 'Chargez la disquette sur le '
      TEXT -'lecteur No 1 '
      TEXT -'<ENTER> ou <BACK> '
ADDRE4 DATA 329,369,520,769,0
TITR4 TEXT 'Le nb de secteurs libres est '
      TEXT -'insuffisant '
      TEXT 'Le fichier complet occupera 227 '
      TEXT -'secteurs '
      TEXT 'Changez votre disquette : '
      TEXT -'<ENTER> '
ADDR4 DATA 520,560,763,0
TIT4 TEXT 'Le fichier complet occupera 227 '
      TEXT -'secteurs '
      TEXT -'Pour les 450 fiches prevues. '
      TEXT -'Pour ouvrir le fichier : <ENTER> '
ADD4 DATA 520,565,764,0
TEXTGR TEXT -'SAISIE DES NOMS DE GROUPES '
      TEXT -'~~~~~'
TEXTG1 TEXT -'*****'
      TEXT -'* '
      TEXT -'* 1- '
      TEXT -'* 2- '
      TEXT -'* 3- '
      TEXT -'* 4- '
      TEXT -'* 5- '
      TEXT -'* 6- '

TEXT -'* 7- * '
TEXT -'* 8- * '
TEXT -'* * '
TEXT -'*****'
TEXTG2 TEXT -'<BACK> terminer '
TEXTG3 TEXT -'<REDO> corriger '
      TEXT -'<S><D> deplacer '
      TEXT -'<ENTER> valider '
ADG3 DATA 623,0
ADGR2 DATA 463,583,823,703,0
ADGR DATA 287,327,402,442,482,522,562,602,642
      DATA 682,722,762,802,842,0
TEXT0 TEXT -'NOM DU DISPOSITIF D''IMPRESSION '
      TEXT -'~~~~~'
ADD0 DATA 404,444,0
NBSAIS DATA 4,28,28,5,20,15,4,6,8,0
ALPHA DATA 0
      EVEN
INIT BL @CLEAR1
      LI R2,TITRE4
      LI R3,ADDR4
      BL @AFFICH
* enter ou back
      BL @ENTER
      CB @RTCLA,@BACK
      JNE INIT1
      B @FICH
* ouverture catalogue disquette
INIT1 LI R1,DSK0
      BL @SATR
      LI R2,15

```

BL @DSR  
\* lecture du catalogue rec:0

BL @INPUT  
BL @DSR

\* fermeture

BL @OVER

\* transfert rec 0 dans tampon

LI R0, TPAB  
LI R1, TAMPON  
LI R2, 38  
BLWP @VMBR

\* longueur du nom dans R3

MOVB \*R1, R3  
SRL R3, 8

\* +20 pour secteurs restants

AI R3, 20

\* transfert des 8 octets dans FAC >834A

LI R0, TPAB  
A R3, R0  
LI R1, >834A  
LI R2, 8  
BLWP @VMBR

\* transformer en nombre simple

BLWP @XMLLNK  
DATA >1200  
MOV @>834A, @SECT

\* comparer avec le minimum

C @SECT, @MINI  
JHE OK

\* si pas assez de secteurs libres

LI R2, TITR4  
LI R3, ADDR4  
BL @AFFICH  
BL @ARRET  
B @INIT

\* nb suffisant

OK LI R2, TIT4  
LI R3, ADD4  
BL @AFFICH  
BL @ARRET

\* ouverture fichier donnees

LI R0, PAB  
LI R1, DSK1  
BL @SATTR  
LI R2, 22  
BLWP @VMBW

\* mode update

LI R0, PAB+1  
LI R1, >0900  
BLWP @VSBW  
BL @DSR

\* saisie nom de groupe

SAISIE BL @CLEAR1  
CLR @NGR  
CLR @NLIG  
CLR @LSAIS  
SETO @DRAP  
CLR @DRAP2  
CLR @LGR  
LI R2, TEXTGR  
LI R3, ADGR  
BL @AFFICH  
LI R2, TEXTG2  
LI R3, ADGR2

BL @AFFICH  
LI R1, >2000  
BL @INITAM

\*\*\*\*\*

\* dans l'enregistrement No 0 :

\* de TAMPON a TAMPON+87 il y a

\* tous les 11 octets les noms

\* des 8 groupes possibles.

\* TRI est a TAMPON+88(2 octets)

\* TOT est a TAMPON+90(2 octets)

\* NGR est a TAMPON+92(2 octets)

\* TAMPON+94=long. nom imprim.

\* TAMPON+95=nom imp (32 octets)

\*\*\*\*\*

\* saisie des groupes

SG0 LI R0, 486  
A @NLIG, R0  
MOV R0, R8  
LI R1, GROUPE  
A @LGR, R1  
MOV R1, R7  
LI R2, 11  
BLWP @VMBW  
MOV R2, @LSAIS  
BL @ACCEPT

\* verifie si ce n'est pas un blanc

MOV R8, R0  
CLR R1  
BLWP @VSBW  
CI R1, >2000  
JEQ SG0

\* compteur de groupe

INC @NGR

\* saisie des lignes suivantes

A @D40, @NLIG  
A @D11, @LGR  
C @NLIG, @D320  
JNE SG0  
CLR @DRAP10

\* saisie nom imprimante

IMPR BL @CLEAR1  
LI R2, TEXT0  
LI R3, ADD0  
BL @AFFICH  
LI R0, 524  
LI R1, NOMIMP+1  
LI R2, 32  
BLWP @VMBW  
MOV R1, R7  
MOV R2, @LSAIS  
CLR @DRAP  
CLR @DRAP2  
BL @ACCEPT  
DEC R2  
SWPB R2  
MOVB R2, @NOMIMP  
MOV @DRAP10, @DRAP10  
JEQ IM

\* Mode update rec 0

BL @CLEAR1  
LI R0, PAB+1  
LI R1, >0900  
BLWP @VSBW  
CLR @NUMENR

BL @NUM  
JMP IM1

\* met le fichier a zero

IM CLR @TRI  
CLR @TOT

\* transfert des saisies

LI R0, TPAB  
LI R1, GROUPE  
LI R2, 94  
BLWP @VMBW  
IM1 LI R0, TPAB+94  
LI R1, NOMIMP  
LI R2, 33  
BLWP @VMBW

\* enregistrement

BL @PRINT  
BL @DSR  
BL @OVER  
MOV @DRAP10, @DRAP10  
JEQ IM2  
BL @DELA1  
B @IMPR1

\* initialisation de "index"

IM2 LI R1, DSK2  
BL @SATTR  
LI R2, 20  
LI R0, PAB  
BLWP @VMBW

\* mode output

LI R0, PAB+1  
LI R1, >0A00  
BLWP @VSBW  
BL @DSR

\* enregistrement 4 octets nuls

LI R0, TPAB  
LI R1, TRI  
LI R2, 4  
BLWP @VMBW  
BL @PRINT  
BL @DSR

\* fermeture fichier

FERM BL @OVER  
B @FICH

X DATA 0

Y DATA 0

XY DATA 0

ASCII DATA 0

SOMME1 DATA 0

SOMME2 DATA 0

SAVRET DATA 0

DRAP3 DATA 0

NUMENR DATA 0

BLANC BYTE 32

POINT BYTE 46

EVEN

\* s/p d'écriture sur dsk

PRINT LI R0, PAB  
LI R1, >0300  
BLWP @VSBW  
RT

\* s/p de lecture sur dsk

INPUT LI R0, PAB  
LI R1, >0200  
BLWP @VSBW

```

RT
* s/p d'addition ascii
ADDIT CLR @ASCII
CLR R1
BLWP @VSB
* on x le 1er octet par 512
* pour qu'il soit significatif
* (on decale l'octet fort de R1 a gauche)
SLA R1,1
MOV R1,R13
INC R0
BLWP @VSB
* on x le 2ieme octet par 16
SRL R1,4
A R1,R13
CLR R14
* plus l'octet suivant
T1 INC R0
BLWP @VSB
* cas d'un espace
CB R1,@BLANC
JNE T2
MOV R0,R14
* cas d'un point
T2 CB R1,@POINT
JEQ T5
SRL R1,8
A R1,R13
MOV R14,R14
JNE T4
* chercher l'espace pour ajouter
* eventuellement le 1er octet prenom
* sauf si mode tri temporaire
MOV @CHTRI,@CHTRI
JEQ T6
LI R2,5
JMP T3
T6 LI R2,15
T3 INC R0
DEC R2
JEQ T5
BLWP @VSB
CB R1,@BLANC
JEQ T4
CB R1,@POINT
JEQ T5
JMP T3
* 1er octet du prenom
T4 INC R0
BLWP @VSB
SRL R1,8
A R1,R13

```

```

T5 MOV R13,@ASCII
RT
* s/p de recherche en bloc
RBLOC MOV R11,@SAVRET
CLR @SOMME2
MOV @TOT,@TOT
JEQ RS2
MOV @TRI,@TRI
JEQ RSEQ
CLR @X
INC @X
MOV @TRI,@Y
* coupe le bloc en deux
DIVISE CLR @XY
A @X,@XY
A @Y,@XY
MOV @XY,R2
SRL R2,1
MOV R2,@XY
JEQ RSEQ
* passer par "index"
MOV R2,@ALPHA
DEC R2
SLA R2,1
MOV @INDEX(R2),@NUMENR
MOV @NUMENR,@SECT
* aller chercher l'enregistrement
BL @NUM
* lecture et addition ascii
BL @INPUT
BL @DSR
LI R0,TPAB+4
BL @ADDIT
C @SOMME2,@ASCII
JEQ RSEQ
MOV @ASCII,@SOMME2
* comparaison avec le nom recherche
C @ASCII,@SOMME1
JEQ RS6
C @X,@Y
JEQ RSEQ
C @ASCII,@SOMME1
JL PART2
* si plus grand que nom recherche
DEC @XY
MOV @XY,@Y
JMP DIVISE
* si plus petit que nom recherche
PART2 INC @XY
MOV @XY,@X
JMP DIVISE
* s/p de recherche sequentielle

```

```

RSEQ C @TOT,@TRI
JEQ RS3
MOV @TRI,@NUMENR
RS1 INC @NUMENR
C @NUMENR,@TOT
JH RS3
MOV @NUMENR,@ALPHA
MOV @NUMENR,@SECT
* no de l'enregistrement
LI R0,PAB+6
LI R1,NUMENR
LI R2,2
BLWP @VMBW
* lecture+addition ascii
BL @INPUT
BL @DSR
LI R0,TPAB+4
BL @ADDIT
* verifie si c'est le bon nom
C @SOMME1,@ASCII
JNE RS1
RS6 MOV @DRAP3,@DRAP3
JNE RS5
JMP RS4
* le fichier est vide
RS2 LI R2,TEXT13
LI R3,ADD14
BL @AFFICH
BL @ENTER
SETO @DRAP
JMP RS4
* pas trouve le nom
RS3 MOV @DRAP3,@DRAP3
JNE RS4
LI R2,TEXT12
LI R3,ADD14
BL @AFFICH
BL @ENTER
SETO @DRAP
RS4 MOV @SAVRET,R11
RT
* nom deja sur fichier
RS5 LI R2,TEXT14
LI R3,ADD14
BL @AFFICH
BL @ENTER
CLR @DRAP3
JMP RS4
*****
* PROGRAMME DE TRI
*****
TEXTE2 TEXT -': ERREUR * '

```

```

TEXT -'<ENTER> pour continuer '
ERR1 TEXT -' Protection contre l'ecriture '
ERR2 TEXT 'Le fichier n'est pas sur la '
TEXT -'disquette '
ERR5 TEXT -' Lecture au-dela du fichier '
ERR6 TEXT 'La disquette n'est pas sur le '
TEXT -'lecteur '
ERRG TEXT ' Sur l'entree/sortie du '
TEXT -'peripherique '
ADER1 DATA 332,567,0
ADER2 DATA 441,0
TEXT1 TEXT -'XXXXXXXXXXXXXXXXXXXXX '
TEXT -'X PROGRAMME DE TRI X '
TEXT -'XXXXXXXXXXXXXXXXXXXXX '
TEXT -'OPTIONS DE TRI '
TEXT -'~~~~~ '
TEXT -'1-Tri alphabetique permanent '
TEXT -'2-Tri temporaire au choix '
TEXT -'3-Retour '
ADTEX1 DATA 48,88,128,251,291,365,405,445,0
TEX2 TEXT -'TRI TEMPORAIRE '
TEXT -'~~~~~ '

```



```

TEXT -'Choix   pour trier '
TEXT -'1      code ptt '
TEXT -'2      ville '
TEXT -'3      cotisation '
TEXT -'4      box '
ADTEX2 DATA 211,251,329,411,451,491,531,571,0
TEXT27 TEXT -'TRI ALPHABETIQUE DES 000 FICHES '
TEXT -'~~~~~'
TEXT -'1-Aucune suppression '
TEXT -'2-Supprimer les effacements '
TEXT -'3-effacements et inactifs '
TEXT -'4-Retour '
ADD27 DATA 243,283,365,405,445,485,0
TEXT28 TEXT '##### POUR LE TRI #####'
TEXT -'# '
TEXT '# '
TEXT -'# '
TEXT '# Le Nb de fiches va de 1 a 000 '

TEXT -'# '
TEXT '# '
TEXT -'# '
TEXT '# '
TEXT -'#####'
TEXT -'# '
TEXTOP TEXT ' ===== OPERATION EN COURS '
TEXT -'===== '
ADD28 DATA 602,642,682,722,762,881,0
TEXT29 TEXT 'OPERATION DE TRI TERMINEE...'
TEXT -'<ENTER> '
ADD29 DATA 882,0
TEXTAL TEXT -'REVENIR TRI ALPHABETIQUE? O/N '
TEXT46 TEXT 'CLASSEMENT "INACTIF" TOUTES '
TEXT -'FICHES '
TEXT ' ETES-VOUS SUR DE VOTRE CHOIX ? '
TEXT -'O/N '
ADD46 DATA 603,682,0
GROFF BYTE 73,78,78,78,78,78,78,78,78

```

```

ERR    BYTE 0
      EVEN
CHTRI  DATA 0
INVER1 DATA 0
INVER2 DATA 0
K      DATA 0
M      DATA 0
ECART  DATA 0
INDEX  BSS 900
STOCK  BSS 1800
* s/p de numerotation
NUM    LI R1,NUM
      LI R0,PAB+6
      LI R2,2
      BLWP @VMBW
      RT
* s/p traitement erreur
ERREUR BL @CLEAR
      LI R0,PAB+1
      BLWP @VSBR
      ANDI R1,>E000
      SRL R1,5
      MOVB R1,@ERR
* fermeture fichier
      LI R0,PAB
      LI R1,ATTRIB
      LI R2,2
      BLWP @VMBW
      LI R9,PAB+9
      MOV R9,@PNTR
      BLWP @DSRLNK
      DATA B
* no d'erreur
      MOVB @ERR,R1
      SRL R1,8
      MOV R1,R5
      CI R5,2
      JNE RR1
      LI R2,ERR2
      JMP ERR3
RR1    CI R5,1
      JNE RR5
      LI R2,ERR1
      JMP ERR3

RR5    CI R5,5
      JNE RR6
      LI R2,ERR5
      JMP ERR3
RR6    CI R5,6
      JNE RRG
      LI R2,ERR6
      JMP ERR3
* affiche erreur
RRG    LI R2,ERRG
      LI R3,ADER2
      BL @AFFICH
      JMP ERR4
ERR3   LI R3,ADER2
      BL @AFFICH
ERR4   LI R2,TEXTE2
      LI R3,ADER1
      BL @AFFICH
      LI R0,341
      MOV R5,R1
      AI R1,48
      SWPB R1
      BLWP @VSBW
BKSCAN BLWP @KSCAN
      CB @RTCLA,@ENT
      JNE BKSCAN
      B @FICH
* ouverture fichier "donnees"
TRIER  LI R1,DSK1
      BL @SATTR
      LI R2,22
      BL @OPEN
* enregistrement 0
      BL @INPUT
      BL @DSR
      BL @READ
* mise en mode update
      LI R0,PAB+1
      LI R1,>0900
      BLWP @VSBW
* options
OPT1   BL @CLEAR
      LI R2,TEX1
      LI R3,ADTEX1

BL @AFFICH
ARR11  BL @ARRET
      CB R4,@UN
      JEQ TRIER1
      CB R4,@DEUX
      JEQ OPT2
      CB R4,@TROIS
      JNE ARR11
      BL @OVER
      B @FICH
* option 2
OPT2   BL @CLEAR1
      SETO @CHTRI
      LI R2,TEX2
      LI R3,ADTEX2
      BL @AFFICH
ARR12  BL @ARRET
      CB R4,@UN
      JNE C2
      LI R8,TPAB+60
      JMP TRIER2
C2     CB R4,@DEUX
      JNE C3
      LI R8,TPAB+65
      JMP TRIER2
C3     CB R4,@TROIS
      JNE C4
      LI R8,TPAB+100
      JMP TRIER2
C4     CB R4,@QUATRE
      JNE ARR12
      LI R8,TPAB+110
      TRIER2 CLR @NUMENR
      SETO @INDX2
      LI R10,STOCK
* saisie des informations a trier
TRI8   INC @NUMENR
      C @NUMENR,@TOT
      JH TRIER3
      BL @NUM
      BL @INPUT
      BL @DSR
* info en somme code ascii
      MOV R8,R0

```

```

BL @ADDIT
* trans. no + ascii dans "stock"
MOV @NUMENR, *R10+
MOV @ASCII, *R10+
JMP TRI8

```

```

TRIER3 B @TRIER4

```

**\* option 1**

```

TRIER1 BL @CLEAR1
CLR @CHTRI
LI R2, TEXT27
LI R3, ADD27
BL @AFFICH
MOV @TOT, R4
LI R2, 266
LI R7, 263
BL @DECI

```

```

ARR6 BL @ARRET
CB R4, @UN
JEQ TRI0
CB R4, @DEUX
JEQ TRI0
CB R4, @TROIS
JEQ TRI0
CB R4, @QUATRE
JNE ARR6
B @OPT1

```

```

TRIO CLR @NUMENR
CLR @INDX2
LI R10, STOCK

```

**\* saisie des noms**

```

TRI1 INC @NUMENR
C @NUMENR, @TOT
JH TRI5
BL @NUM
BL @INPUT
BL @DSR

```

**\* options**

```

CB R4, @UN
JEQ TRI4
LI R0, TPAB+118
BLWP @VSBW
CB R4, @TROIS
JNE TRI6
CB R1, @I
JEQ TRI2

```

```

TRI6 CB R1, @E
JNE TRI4

```

```

TRI7 C @NUMENR, @TOT
JNE TRI2
DEC @TOT
JMP TRI1

```

**\* dernier enregis. a la place de  
\* celui qui est marque "e" ou "i"**

```

TRI2 LI R0, PAB+6
LI R1, TOT
LI R2, 2
BLWP @VMBW
BL @DSR

```

**\* verif. si "e" ou "i" lui aussi**

```

LI R0, TPAB+118
BLWP @VSBW
CB R1, @E
JEQ TRI3
CB R1, @I

```

```

JEQ TRI3
* prends la place du "e" ou "i"

```

```

BL @NUM
BL @PRINT
BL @DSR
DEC @TOT
JMP TRI4
TRI3 DEC @TOT
JMP TRI7

```

**\* nom en somme code ascii**

```

TRI4 LI R0, TPAB+4
BL @ADDIT

```

**\* trans. no + ascii dans "stock"**

```

MOV @NUMENR, *R10+
MOV @ASCII, *R10+
JMP TRI1

```

**\* enregistrement de rec 0**

```

TRI5 MOV @TOT, @TRI
BL @WRITE
CLR @NUMENR
BL @NUM
BL @PRINT
BL @DSR

```

**\* fermeture du fichier "donnees"**

```

TRIER4 BL @OVER
*****

```

**\* chaque enreg. valide est dans**

**\* stock sur 4 octets.**

**\* 2 pour le no d'enregistrement**

**\* 2 pour la somme ASCII**

\*\*\*\*\*

```

LI R2, TEXT28
LI R3, ADD28
BL @AFFICH
LI R2, 712
LI R7, 709
MOV @TOT, R4
BL @DECI

```

**\* tri numerique sommes ascii**

**\* methode shell-metzner**

```

MOV @TOT, @ECART
TRIA1 CLR @X
MOV @TOT, @K

```

**\* coupe en deux, sortie si=0**

```

MOV @ECART, R15
SRL R15, 1
MOV R15, @ECART
CI R15, 1
JL INDENR
INC @X
S @ECART, @K

```

```

TRIA2 MOV @X, @Y

```

```

TRIA3 MOV @Y, @M

```

```

A @ECART, @M

```

**\* reperege enr. Y dans "stock"**

```

LI R1, STOCK+2
MOV @Y, R12
DEC R12
SLA R12, 2
A R12, R1
MOV *R1, @SOMME1

```

**\* reperege enr. M dans "stock"**

```

LI R2, STOCK+2
MOV @M, R12

```

```

DEC R12

```

```

SLA R12, 2

```

```

A R12, R2

```

```

MOV *R2, @SOMME2

```

**\* comparaison des deux enr.**

```

C @SOMME1, @SOMME2

```

```

JLE TRIA4

```

**\* inversion des enregistrements**

```

DECT R1
MOV *R1+, @INVER1
MOV *R1, @INVER2
DECT R1
DECT R2
MOV *R2+, *R1+
MOV *R2, *R1
DECT R2
MOV @INVER1, *R2+
MOV @INVER2, *R2

```

**\* suite du tri**

```

S @ECART, @Y
JH TRIA3
TRIA4 INC @X
C @X, @K
JH TRIA1
JMP TRIA2

```

**\* initialisation "index"**

```

INDENR LI R1, STOCK
LI R2, INDEX
CLR R3
IND1 MOV *R1+, *R2+
INC R3
C R3, @TOT
JH IND2
INCT R1
JMP IND1

```

**\* ouverture fichier "index"**

```

IND2 MOV @CHTRI, @CHTRI
JNE IND4+4
CLR @LSAIS
CLR @NUMENR
LI R1, DSK2
BL @SATTR
LI R2, 20
BL @OPEN

```

**\* mise en mode output**

```

LI R0, PAB+1
LI R1, >0A00
BLWP @VSBW

```

**\* enregistrement de "index"**

```

CLR R12
LI R1, INDEX
IND3 LI R0, TPAB
LI R2, 254
BLWP @VMBW
A R2, R12
SRL R2, 1
A R2, @LSAIS
BL @PRINT
BL @DSR
C @LSAIS, @TOT
JHE IND4

```

**\* enregistrement suivant**

```

INC @NUMENR
LI R0, PAB+6

```

```

LI R1,NUMENR          BLWP @VSBW          C @NUMENR,@TOT
LI R2,2               LI R2,TEXT46        JH FINAPP
BLWP @VMBW           LI R3,ADD46         BL @NUM
LI R1,INDEX          BL @AFFICH         BL @INPUT
A R12,R1             APP BL @ARRET       BL @DSR
JMP IND3
* operation terminee
IND4 BL @OVER
LI R2,TEXT29
LI R3,ADD29
BL @AFFICH           APP1 LI R2,TEXTOP
BL @ENTER           LI R3,ADD46+2
CLR R12             BL @AFFICH
B @FICH             * mode update
.....
* supprime l'appartenance
.....
APPGR LI R0,568
LI R1,>3E00          LOOP1 INC @NUMENR

```

```

C @NUMENR,@TOT
JH FINAPP
BL @NUM
BL @INPUT
BL @DSR
* charge inactif + 8 fois non
LI R0,TPAB+118
LI R1,GROFF
LI R2,9
BLWP @VMBW
* reenregistrement
BL @NUM
BL @PRINT
BL @DSR
JMP LOOP1
* termine
FINAPP BL @OVER
BL @CLEAR1
B @FINI2

```

## Source "FICH3"

```

TEXT30 TEXT '-XXXXXXXXXXXXXXXXXXXXXXXXX '
TEXT '-X IMPRESSION DU FICHIER X '
TEXT '-XXXXXXXXXXXXXXXXXXXXXXXXX '
TEXT '-OPTIONS D'IMPRESSION '
TEXT '-~~~~~ '
TEXT '-1 Fichier organise '
TEXT '-2 Etiquettes '
TEXT '-3 Retour '
ADD30 DATA 47,87,127,369,409,490,530,570,0
ENTET1 TEXT ' No NOM PRENOM
TEXT ' ADRESSE'
TEXT ' CODE VILLE
TEXT '
TEXT ' TEL BOX
TEXT 'COTISATION '
TEXT '
ENTET2 TEXT '--- ----- '
TEXT '-----'
TEXT '----- ---- '
TEXT '-----'
TEXT '--- ----- '
TEXT '----- '
NET TEXT '
TEXT '
TEXT31 TEXT '***** IMPRESSION EN COURS *****'
TEXT '-* '
TEXT32 TEXT ' VOULEZ-VOUS UN TITRE ? O/N
TEXT '- '
TEXT33 TEXT ' AVEC DES GROS CARACTERES? O/N'
TEXT '- '
TEXT34 TEXT 'XXXXXX IMPRESSION TERMINEE XXXXX'
TEXT '-X '
TEXT '-<ENTER> '
ADD34 DATA 683,776,0
TEXT35 TEXT '+ centrage + centrage +'
TEXT '- '
ADD35 DATA 683,0
TEXT36 TEXT ' VOULEZ-VOUS UN SOUS TITRE? O/N'
TEXT '- '
TEXT37 TEXT ' ALLUMEZ L'IMPRIMANTE ... <ENTER'
TEXT '-> '

```

```

TEXT38 TEXT 'VOUS SELECTIONNEZ DES GROUPES ? O/'
TEXT '-N '
TEXT39 TEXT ' CHOISISSEZ O/N POUR CHAQUE GROUP'
TEXT '-E '
MODIMP TEXT '- MODIFICATION DU NOM? O/N '
SELECT TEXT ' IMPRESSION SELECTIVE PAR No? O/'
TEXT '-N '
NUMERO TEXT ' No DE FICHE: 000 (ordre alpha'
TEXT '-) '
NOUVEL TEXT '-UN NOUVEAU CHAPITRE ? O/N '
ADDTER DATA 932,0
ININUM DATA >3030,>3000
* caracteres compressees
COMPR DATA >1B43
* caracteres dilates et gras
BIGCAR DATA >1B0E,>1B23
* fin caracteres dilates,gras+mode normal
FINBIG DATA >1B0F,>1B24,>1B4E
TIMP EQU >1200
DRAP10 DATA 0
INDX2 DATA 0
CHOIXG DATA >4E4E,>4E4E,>4E4E,>4E4E
EVEN
TLIGNE BSS 136
IMP DATA >0012,>1200,>0088,>0000
BYTE >00
NOMIMP BYTE >00
DATA >2020,>2020,>2020,>2020,>2020,>2020
DATA >2020,>2020
DATA >2020,>2020,>2020,>2020,>2020,>2020
DATA >2020,>2020
* s/p lecture informations
READ LI R0,TPAB
LI R1,GROUPE
LI R2,94
BLWP @VMBR
LI R0,TPAB+94
LI R1,NOMIMP
LI R2,33
BLWP @VMBR
RT

```

# \* s/p saisie info ttab

```
TRANS BLWP @VSBW
      CB R1,@POINT
      JEQ RET4
      MOVB R1,*R3+
      INC R0
      DEC R2
      JEQ RET4
      JMP TRANS
```

RET4 RT

# \* s/p efface tligne

```
CLSTL CLR R2
      LI R1,>2000
BLANC1 MOVB R1,@TLIGNE(R2)
      INC R2
      CI R2,136
      JNE BLANC1
      RT
```

# \* verif. "index" deja charge

```
IMPRES SETO @DRAP10
      MOV @INDX2,@INDX2
      JNE IMPR1
      B @GERER
```

# \* ouverture fichier "donnees"

```
IMPR1 LI R1,DSK1
      BL @SATTR
      LI R2,22
      BL @OPEN
```

# \* saisie enr.0

```
CLR @NUMENR
BL @INPUT
BL @DSR
BL @READ
```

# \* nom imprimante

```
BL @CLEAR1
LI R2,TEXT0
LI R3,ADD0
BL @AFFICH
LI R1,NOMIMP+1
LI R2,32
LI R0,524
BLWP @VMBW
LI R2,MODIMP
LI R3,ADD14
BL @AFFICH
ARR16 BL @ARRET
      CB R4,@NON
      JEQ AR0
```

# \* modif du nom

```
CB R4,@OUI
JNE ARR16
CLR @DRAP1
CLR @DRAP2
SETO @DRAP4
BL @CLS24
CLR R12
B @IMPR+4
AR0 BL @CLEAR
      CLR @DRAP4
      LI R2,TEXT30
      LI R3,ADD30
      BL @AFFICH
ARR7 BL @ARRET
      CB R4,@DEUX
```

```
JNE AR
B @ETIQ
AR CB R4,@UN
      JEQ AR2
      CB R4,@TROIS
      JNE ARR7
      CLR @DRAP10
      BL @OVER
      B @FICH
```

# \* efface option 2 et 3

```
AR2 CLR R12
      CLR R15
      LI R0,530
      LI R1,NET
      LI R2,12
      BLWP @VMBW
      LI R0,570
      LI R1,NET
      LI R2,8
      BLWP @VMBW
```

# \* impression selective

```
LI R2,SELECT
LI R3,ADD35
BL @AFFICH
STOP BL @ARRET
      CB R4,@NON
      JEQ OPGR
      CB R4,@OUI
      JNE STOP
      SETO @DRAP6
      JMP AR8
```

# \* option groupes

```
OPGR LI R2,TEXT38
      LI R3,ADD35
      BL @AFFICH
      CLR @DRAP6
ARR10 BL @ARRET
      CB R4,@NON
      JEQ AR8
      CB R4,@OUI
      JNE ARR10
      SETO R15
CHGR LI R2,TEXT39
      LI R3,ADD35
      BL @AFFICH
```

```
CLR R8
CLR @LGR
AR7 LI R0,770
      LI R1,GROUPE
      A @LGR,R1
      LI R2,11
      BLWP @VMBW
      SETO @DRAP2
      SETO @DRAP4
      LI R0,782
      LI R2,1
      MOV R2,@LSAIS
      LI R7,CHOIXG
      A R8,R7
      BL @ACCEPT
      A @D11,@LGR
      INC R8
      C R8,@NGR
      JL AR7
```

```
AR8 LI R0,760
      LI R1,NET
      LI R2,40
      BLWP @VMBW
      LI R2,TEXT37
      LI R3,ADD35
      BL @AFFICH
      BL @ENTER
      MOV @DRAP11,@DRAP11
      JEQ AR11
      B @PLACE
```

# \* ouvre l'imprimante

```
AR1 B @IMPRO
ARR11 LI R1,IMP
      BL @SATTR
      LI R2,42
      BL @OPEN
ARR12 LI R2,TEXT32
      LI R3,ADD35
      BL @AFFICH
ARR8 BL @ARRET
      CB R4,@NON
      JEQ AR1
      CB R4,@OUI
      JNE ARR8
      LI R2,TEXT33
      LI R3,ADD35
      BL @AFFICH
      CLR R14
ARR9 BL @ARRET
      CB R4,@NON
      JEQ AR3
      CB R4,@OUI
      JNE ARR9
```

# \* caracteres doubles et gras

```
LI R0,TIMP
LI R1,BIGCAR
LI R2,4
BLWP @VMBW
LI R0,PAB+5
LI R1,>0400
BLWP @VSBW
BL @PRINT
BL @DSR
```

# \* saisie titre

```
AR3 BL @OVER
      LI R2,TEXT35
      LI R3,ADD35
      BL @AFFICH
      BL @CLSTL
      LI R2,40
      MOV R2,@LSAIS
      LI R0,760
      LI R7,TLIGNE
      SETO @DRAP4
      CLR @DRAP2
      BL @ACCEPT
      DEC R2
      MOV R2,R13
```

# \* reouvre dis/var(R2)

```
LI R0,PAB+5
MOV R2,R1
SWPB R1
BLWP @VSBW
```



```

BL @DSR
* impression
LI R0,TIMP
LI R1,TLIGNE
BLWP @VMBW
BL @PRINT
BL @DSR
MOV R14,R14
JNE AR9

```

```

* soulignement
LI R0,TIMP
AR4 BLWP @VSBW
CB R1,@BLANC
JNE AR5
INC R0
JMP AR4
AR5 LI R1,>7E00
AR6 BLWP @VSBW
INC R0
DEC R13
JNE AR6
BL @PRINT
BL @DSR

```

```

* lignes blanches
BL @CLSTI
BL @CLSTP
LI R1,1
BL @SAUTLI+4
BL @DSR
CB R4,@NON
JEQ IMPRO

```

```

* fin gros caracteres gras
LI R0,TIMP
LI R1,FINBIG
LI R2,4
BLWP @VMBW
LI R0,PAB+5
LI R1,>0400
BLWP @VSBW
BL @PRINT
BL @DSR
BL @OVER

```

```

* reouverture mode dis/var 136
LI R1,IMP
LI R2,42
BL @OPEN

```

```

* sous titre
IMPRO LI R0,760
LI R1,NET
LI R2,40
BLWP @VMBW
LI R2,TEXT36
LI R3,ADD35
BL @AFFICH
BL @ARRET
CB R4,@NON
JEQ AR9
SETO R14
B @AR3+4
AR9 LI R1,NET
LI R0,680
LI R2,40
BLWP @VMBW
LI R2,TEXT31

```

```

LI R3,ADD35
BL @AFFICH
LI R0,760
LI R1,NET
LI R2,40
BLWP @VMBW
* caracteres compresses 136/ligne
LI R0,TIMP
LI R1,COMPR
LI R2,2
BLWP @VMBW
LI R0,PAB+5
LI R1,>0200
BLWP @VSBW
BL @PRINT
BL @DSR

```

```

* imprime l'en tete
LI R0,PAB+5
LI R1,>8800
BLWP @VSBW
LI R3,2
LI R0,TIMP
LI R1,ENTET1
LI R2,136
ENT2 BLWP @VMBW
BL @DSR
LI R1,ENTET2
DEC R3
JNE ENT2
BL @INIDSK
CLR @LGR

```

```

* saisie par numero
MOV @DRAP6,@DRAP6
JEQ TRITEM
SETO R12
LI R2,NUMERO
LI R3,ADD35
BL @AFFICH
LI R2,TEXTG2
LI R3,ADDTER
BL @AFFICH
MOUV LI R2,ININUM
LI R3,TLIGNE
MOUV1 MOVB *R2+,*R3+
JNE MOUV1
CLR @DRAP2
SETO @DRAP4
LI R0,699
LI R7,TLIGNE
LI R2,3
MOV R2,@LSAIS
BL @ACCEPT
CI R2,3
JNE MOUV

```

```

* transforme ascii en hexa
LI R2,-48
SWPB R2
LI R0,TLIGNE
CLR R1
MOVB *R0+,R1
AB R2,R1
MOV R1,@NUMENR
MOVB *R0+,R1
AB R2,R1

```

```

SRL R1,4
A R1,@NUMENR
AB R2,*R0
CLR R1
MOVB *R0,R1
SWPB R1
A R1,@NUMENR
MOV @NUMENR,@NUMENR
JEQ MOUV
JMP AUTRE+4

```

```

* si tri temporaire
TRITEM C @TRI,@TOT
JEQ AUTRE
MOV @CHTRI,@CHTRI
JEQ AUTRE
MOV @TRI,@LSAIS
MOV @TOT,@TRI

```

```

* saisie enregistrements 1 a tot
AUTRE INC @NUMENR
C @NUMENR,@TOT
JLE IMPR2
MOV @CHTRI,@CHTRI
JEQ AUTR1
MOV @LSAIS,@TRI
AUTR1 B @FINIMP
IMPR2 BL @CLSTL
BL @PASS
IMPR3 BL @INPUT
BL @DSR

```

```

* controle option groupe
MOV R15,R15
JEQ IMPR5
LI R0,TPAB+118
LI R1,BUFGR
LI R2,9
BLWP @VMBW
CLR @NLIG
LI R1,BUFGR+1
LI R2,CHOIXG
CONTR CB *R2,@OUI
JEQ CONTR1
INC R2
INC R1
JMP CONTR2
CONTR1 CB *R2+,*R1+
JEQ IMPR5
CONTR2 INC @NLIG
C @NLIG,@NGR
JL CONTR
MOV @DRAP11,@DRAP11
JNE IMPR6
JMP AUTRE
IMPR6 B @LOOP
IMPR7 B @PLACE1

```

```

* no dans tligne
IMPR5 MOV @DRAP11,@DRAP11
JNE IMPR7
INC @LGR
MOV @LGR,R4
LI R2,TIMP+2
LI R7,TIMP-1
BL @DECI
LI R0,TIMP
LI R1,TLIGNE

```

LI R2,3	LI R3,TLIGNE+115	CLR @DRAP2
BLWP @VMBR	LI R2,4	CLR @DRAP4
<b>* nom prenom</b>	BL @TRANS	CLR @DRAP6
LI R0,TPAB+4	<b>* no de licence</b>	CLR @DRAP10
LI R3,TLIGNE+5	LI R0,TPAB+104	B @FICH
LI R2,28	LI R3,TLIGNE+121	<b>* nouveau chapitre?</b>
BL @TRANS	LI R2,6	NOUVE LI R2,NOUVEL
<b>* adresse</b>	BL @TRANS	LI R3,ADDRET
LI R0,TPAB+32	<b>* tligne dans timp</b>	BL @AFFICH
LI R3,TLIGNE+33	LI R0,TIMP	STOP1 BL @ARRET
LI R2,28	LI R1,TLIGNE	CB R4,@NON
BL @TRANS	LI R2,136	JEQ FINIMP
<b>* code postal</b>	BLWP @VMBW	CB R4,@OUI
LI R0,TPAB+60	<b>* init pab imprimante</b>	JNE STOP2
LI R3,TLIGNE+62	BL @INIIMP	BL @CLS24
LI R2,5	BL @SATTR	CLR R12
BL @TRANS	BL @PRINT	BL @REMISE
<b>* ville</b>	BL @DSR	B @AR12
LI R0,TPAB+65	<b>* reinit pab fichier</b>	<b>* s/p mode normal imprimante</b>
LI R3,TLIGNE+68	BL @INIDSK	REMISE MOV R11,@SAVRET
LI R2,20	BL @SATTR	BL @INIIMP
BL @TRANS	MOV R12,R12	BL @SATTR
<b>* tel</b>	JNE AUTR2	LI R0,FINBIG+2
LI R0,TPAB+85	B @AUTRE	LI R1,NORM
LI R3,TLIGNE+89	AUTR2 B @MOUV	LI R2,4
LI R2,15	FINIMP BL @OVER	BLWP @VMBW
BL @TRANS	BL @CLS24	LI R0,PAB+5
<b>* box</b>	BL @REMISE	LI R1,>0400
LI R0,TPAB+110	BL @OVER	BLWP @VSBW
LI R3,TLIGNE+105	FINI2 LI R2,TEXT34	BL @PRINT
LI R2,8	LI R3,ADD34	BL @DSR
BL @TRANS	BL @AFFICH	MOV @SAVRET,R11
<b>* cotisation</b>	BL @ENTER	RT
LI R0,TPAB+100	CLR R12	

\*\*\*\*\*  
**\* ETIQUETTES**  
 \*\*\*\*\*

```

TEXT40 TEXT -'ETIQUETTES D''ADRESSES '
        TEXT -'~'
        TEXT -'MISE EN PLACE '
        TEXT -'1-pas de modification '
        TEXT -'2-modification '
        TEXT -'3-retour '
ADD40 DATA 369,409,533,609,649,689,0
TEXT41 TEXT -'GROUPES A IMPRIMER '
        TEXT -'~'
ADD41 DATA 450,490,0
TEXT42 TEXT -'MODIFICATION DE MISE EN PLACE '
        TEXT -' Marge de gauche : '
        TEXT -' Ecart central : '
        TEXT -' Sauts de lignes : '
        TEXT -'Nb d''etiqligne : '
ADD42 DATA 525,609,649,685,731,0
FORM1 TEXT 'TITRE..NOM...PRENOM.....'
FORM2 TEXT 'ADRESSE.....'
FORM3 TEXT 'CODE.POSTAL....VILLE.....'
TEXT43 TEXT -'LIGNE D''IMPRESSION TROP LONGUE '
TEXT44 TEXT -' EMBLACEMENT CORRECT? O/N '
TEXT45 TEXT ' IMPRESSION CARATERES GRAS? O/'
        TEXT -'N '
* code imprimante pour saut de ligne [CHR$(10)]
HEX10 DATA >0A0A,>0A0A,>0A0A,>0A0A,>0A0A

```

```

DRAP11 DATA 0
* options etiquettes*
MG DATA 0
ENTRE DATA 4
SAUT DATA 5
NBETIQ DATA 2
*****
TOTAL DATA 72
EMPLAC DATA 0
* nb de colonnes de l'imprimante
MAXI DATA 80
        EVEN
* s/p impression de la ligne
IMPLIG MOV R11,@SAVRET
        LI R0,TIMP
        LI R1,TLIGNE
        MOV @TOTAL,R2
        BLWP @VMBW
        BL @PRINT
        BL @DSR
        BL @CLSTL
        MOV @SAVRET,R11
        RT
* s/p init pab imprimante
INIIMP LI R0,PAB
        LI R1,IMP
        LI R2,42
        BLWP @VMBW

```

```

RT
* s/p init pab disquette
INIDSK LI R0,PAB
        LI R1,DSK1
        LI R2,22
        BLWP @VMBW
        RT
* s/p mode dis/var (total)
DISVAR LI R0,PAB+5
        MOV @TOTAL,R1
        SWPB R1
        BLWP @VSBW
        RT
* s/p sauts de lignes
SAUTLI MOV @SAUT,R1
        LI R0,PAB+5
        MOV R1,R2
        SWPB R1
        BLWP @VSBW
        LI R0,TPAB
        LI R1,HEX10
        BLWP @VMBW
        RT
* s/p de placement
PLACER LI R0,TIMP
        A @MG,R0
        BLWP @VMBW
        A R2,R0

```

```

A @ENTRE,R0
BLWP @VMBW
RT
* s/p efface timp
CLSTP LI R0,TIMP
LI R1,TLIGNE
LI R2,80
BLWP @VMBW
RT
ETIQ BL @CLEAR1
LI R2,TEXT40
LI R3,ADD40
BL @AFFICH
ARR13 BL @ARRET
CB R4,@UN
JEQ ETIQ2
CB R4,@DEUX
JEQ ETIQ1
CB R4,@TROIS
JNE ARR13
B @ARO
* modif de mise en place
ETIQ1 LI R2,TEXT42
LI R3,ADD42
BL @AFFICH
SETO @DRAP2
SETO @DRAP4
SETO @DRAP6
MG1 CLR @NLIG
CLR @LGR
LI R6,48
A R6,@MG
A R6,@ENTRE
A R6,@SAUT
A R6,@NBETIQ
LI R5,4
LI R2,1
MOV R2,@LSAIS
MG2 LI R0,629
A @NLIG,R0
LI R1,MG+1
A @LGR,R1
MOV R1,R7
BL @ACCEPT
A @D40,@NLIG
INCT @LGR
DEC R5
JNE MG2
* tranforme en nombre
LI R6,-48
A R6,@MG
A R6,@ENTRE
A R6,@SAUT
A R6,@NBETIQ
MOV @D33,@TOTAL
MOV @NBETIQ,R2
M0 DEC R2
JEQ M1
A @D33,@TOTAL
JMP M0
M1 A @MG,@TOTAL
A @ENTRE,@TOTAL
C @TOTAL,@MAXI
JLE ETIQ2

```

#### \* impossible

```

LI R2,TEXT43
LI R3,ADD14
BL @AFFICH
BL @ARRET
BL @CLS24
JMP MG1

```

#### \* choix des groupes

```

ETIQ2 BL @CLEAR1
LI R2,TEXT41
LI R3,ADD41
BL @AFFICH
SETO @DRAP11
CLR @DRAP6
B @CHGR

```

#### \* ouvrir l'imprimante

```

PLACE LI R1,IMP
LI R2,43
BL @OPEN
LI R2,TEXT45
LI R3,ADD35
BL @AFFICH
ARR14 BL @ARRET
CB R4,@NON
JEQ MG3
CB R4,@OUI
JNE ARR14

```

#### \* init caracteres gras

```

LI R0,TIMP
LI R1,BIGCAR+2
LI R2,2
BLWP @VMBW
LI R0,PAB+5
LI R1,>0200
BLWP @VSBW
BL @PRINT
BL @DSR
MG3 BL @DISVAR

```

#### \* etiquette test

```

BL @CLSTL
BL @CLSTP
LI R1,FORM1
LI R2,33
BL @PLACER
LI R2,TEXT31
LI R3,ADD35
BL @AFFICH

```

#### \* 1ere ligne

```

BL @PRINT
BL @DSR

```

#### \* 2ieme ligne

```

BL @CLSTP
LI R1,FORM2
LI R2,33
BL @PLACER
BL @DSR

```

#### \* 3ieme ligne

```

BL @CLSTP
LI R1,FORM3
LI R2,33
BL @PLACER
BL @DSR

```

#### \* sauts

```

MOV @SAUT,@SAUT

```

#### JEQ CORR

```

BL @SAUTLI
BL @DSR
BL @DISVAR
CORR LI R2,TEXT44
LI R3,ADD14
BL @AFFICH
ARR15 BL @ARRET
CB R4,@OUI
JEQ MG4
CB R4,@NON
JNE ARR15
BL @CLEAR1
B @ETIQ1
MG4 CLR @DRAP2
CLR @DRAP4
CLR @DRAP6
CLR @NUMENR
CLR @LSAIS
MOV @NBETIQ,R13
BL @CLS24
BL @CLSTL
LI R2,TEXT31
LI R3,ADD35
BL @AFFICH
CLR R8

```

#### \* enregistrements

```

LOOP INC @NUMENR
C @NUMENR,@TOT
JH L1
BL @INIDSK
BL @SATTR
SETO R15
BL @NUM
B @IMPR3
L1 CI R13,1
JEQ PLACE0
B @FIN

```

#### \* stockage

```

PLACE0 SETO R8
BL @CLSTL
LI R0,TPAB
LI R1,TLIGNE
LI R2,127
BLWP @VMBW
PLACE1 LI R1,STOCK
A @LSAIS,R1
LI R2,118
LI R0,TPAB
BLWP @VMBR
A R2,@LSAIS
DEC R13
JNE LOOP
BL @INIIMP
BL @SATTR
BL @DISVAR

```

#### \* titre

```

PLACE2 LI R3,STOCK
MOV R3,@EMPLAC
LI R4,TLIGNE
A @MG,R4
MOV @NBETIQ,R5
PLACE3 LI R2,4
PL1 CB *R3,@POINT

```

JEQ PLACE4	PL5	CB *R3,@POINT	PL9	CB *R3,@POINT
MOVB *R3+,*R4+	JEQ PL6		JEQ PL10	
DEC R2	MOVB *R3+,*R4+		MOVB *R3+,*R4+	
JNE PL1	DEC R2		DEC R2	
<b>* nom prenom</b>	JNE PL5		JNE PL9	
PLACE4 LI R2,28	PL6 DEC R5		PL10 DEC R5	
INC R4	JEQ PL7		JEQ PL11	
MOV @EMPLAC,R3	LI R3,STOCK+150		LI R4,TLIGNE+33	
AI R3,4	LI R4,TLIGNE+33		A @ENTRE,R4	
PL2 CB *R3,@POINT	A @ENTRE,R4		LI R3,STOCK+178	
JEQ PL3	JMP PLACE5		MOV R3,@EMPLAC	
MOVB *R3+,*R4+	PL7 BL @IMPLIG		JMP PLACE5	
DEC R2	<b>* code et ville</b>		PL11 BL @IMPLIG	
JNE PL2	LI R3,STOCK+60		<b>* sauts</b>	
PL3 DEC R5	MOV R3,@EMPLAC		MOV @SAUT,@SAUT	
JEQ PL4	LI R4,TLIGNE		JEQ SUIVRE	
LI R4,TLIGNE+33	A @MG,R4		BL @SAUTLI	
A @ENTRE,R4	MOV @NBETIQ,R5		BL @DSR	
LI R3,STOCK+118	PLACE6 LI R2,5		BL @DISVAR	
MOV R3,@EMPLAC	PL8 CB *R3,@POINT		SUIVRE CLR @LSAIS	
JMP PLACE3	JEQ PLACE7		MOV @NBETIQ,R13	
PL4 BL @IMPLIG	MOVB *R3+,*R4+		MOV R8,R8	
<b>* adresse</b>	DEC R2		JNE FIN	
LI R3,STOCK+32	JNE PL8		B @LOOP	
LI R4,TLIGNE	PLACE7 LI R2,20		FIN BL @INIDSK	
A @MG,R4	INC R4		BL @SATTR	
MOV @NBETIQ,R5	MOV @EMPLAC,R3		CLR @DRAP11	
PLACE5 LI R2,28	AI R3,5		B @FINIMP	

## Source "FICH4"

\*\*\*\*\*

### \* MODIFIER UN NOM

\*\*\*\*\*

#### \* s/p d'attente

```

DELA1 CLR R15
      INC R15
      JNE DELAI+2
      RT

DRAP4 DATA 0
TEXT11 TEXT -' modifier une fiche '
      TEXT -'Rentrez le nom de la personne '
      TEXT -'que vous voulez modifier. '
      TEXT -'NOM Prenom: '

ADD11 DATA 48,325,367,520,0
TEXT12 TEXT 'NOM INCONNU AU FICHIER!...<ENTER>'
      TEXT -'> '

TEXT13 TEXT -'LE FICHIER EST VIDE!...<ENTER> '
TEXT14 TEXT -'NOM DEJA SUR FICHIER!...<ENTER> '
ADD14 DATA 925,0
TEXT15 TEXT -'UNE AUTRE MODIFICATION ? O/N '
TEXTI TEXT -'inactif '
ADDI DATA 229,0
TEXTA TEXT -' actif '
TEXTRI TEXT 'SI VOUS AVEZ MODIFIE LE NOM OU LE'
      TEXT -' PRENOM '
      TEXT 'D''UNE PERSONNE, FAITES FONCTIONNE'
      TEXT -'R LE '
      TEXT -'PROGRAMME DE TRI IMPERATIVEMENT. '
      TEXT -'<ENTER> '

ADDTRI DATA 520,601,684,775,0
TEXTTEX TEXT 'Dans MODIFIER,EFFACER,AFFICHER, '

```

```

TEXT -'vous '
TEXT 'pouvez, pour la saisie, ne mettre '
TEXT -'que '
TEXT -'les 3 premieres lettres du nom '
TEXT -'la premiere lettre du prenom '
TEXT 'sans oublier l''espace entre '
TEXT -'nom-prenom '
TEXT -'<ENTER> '

```

```

ADDEX DATA 441,481,521,561,601,775,0
EVEN

```

\*\*\*\*\*

```

MODIF BL @CLEAR1
      LI R2,TEXTTEX
      LI R3,ADDEX
      BL @AFFICH
      BL @ENTER

MODIFO BL @CLEAR1
      LI R2,TEXT11
      LI R3,ADD11
      BL @AFFICH

```

#### \* saisie du nom

```

MODIF1 LI R1,>2E00
      BL @INITAM
      LI R0,532
      LI R1,TAMPON+4
      MOV R1,R7
      LI R2,28
      BLWP @VMBW
      MOV R2,@LSAIS
      CLR @DRAP2
      BL @ACCEPT

```



### \* transfert dans tpab

```
LI R0, TPAB+4
LI R1, TAMPON+4
LI R2, 28
BLWP @VMBW
```

### \* somme codes ascii

```
BL @ADDIT
MOV @ASCII, @SOMME1
CLR @DRAP3
```

### \* recherche dans fichier

```
BL @RBLOC
MOV @DRAP, @DRAP
JNE AUTMOD
```

### \* transfert dans "tampon"

```
LI R0, TPAB
LI R1, TAMPON
LI R2, 127
BLWP @VMBR
```

### \* verifie si effacement

```
MOV @DRAP7, @DRAP7
JNE MOD4
MOVB @BUFG, R3
CB R3, @E
JNE MODIF2
BL @CLEAR1
LI R2, TEXT18
LI R3, ADD18
BL @AFFICH
ARR3 BL @ARRET
CB R4, @NON
JEQ AUTMOD
CB R4, @OUI
JNE ARR3
```

### \* reintroduction

```
LI R0, TPAB+118
MOVB @A, R1
BLWP @VSBW
SETO @DRAP4
B @AJENR4
```

### \* affichage

```
MODIF2 MOV @DRAP5, @DRAP5
JNE MOD3
BL @CLEAR1
LI R2, TEXT6
LI R3, ADD6
BL @AFFICH
```

### \* actif ou inactif

```
MOVB @BUFG, R4
CB R4, @I
JNE ACTIF
LI R2, TEXTI
LI R3, ADDI
BL @AFFICH
JMP NUM1
ACTIF LI R2, TEXTA
LI R3, ADDI
BL @AFFICH
NUM1 MOV @ALPHA, R4
SETO @DRAP4
CLR @DRAP3
B @AJOUT4
```

### \* autre modification ?

```
AUTMOD BL @CLS24
MOV @DRAP5, @DRAP5
```

```
JNE MOD1
MOV @DRAP7, @DRAP7
JNE MOD5
LI R2, TEXT15
LI R3, ADD14
BL @AFFICH
CLR @DRAP
CLR @DRAP2
BL @DELA1
ARR1 BL @ARRET
CB R4, @OUI
JEQ MOD2
CB R4, @NON
JNE ARR1
BL @CLEAR1
LI R2, TEXTRI
LI R3, ADDTRI
BL @AFFICH
BL @ENTER
B @OPT
MOD1 B @AUTEFF
MOD2 B @MODIF0
MOD3 B @EFA1
MOD4 MOV @ALPHA, @NUMENR
B @RECH2
MOD5 B @RECH6
```

ARR1

MOD1

MOD2

MOD3

MOD4

MOD5

### \* EFFACER UN NOM

```
*****
DRAP5 DATA 0
```

```
TEXT16 TEXT -' effacer une fiche '
TEXT -'Rentrez le nom de la personne '
TEXT -'que vous voulez effacer. '
TEXT -'NOM Prenom: '
```

```
ADD16 DATA 48,365,407,520,0
```

```
TEXT17 TEXT -' UN AUTRE EFFACEMENT ? O/N '
TEXT18 TEXT -'LA FICHE EST MARQUEE EFFACEMENT '
TEXT -'REINTRODUCTION ? O/N '
```

```
ADD18 DATA 525,610,0
```

EVEN

\*\*\*\*\*

```
EFFACE BL @CLEAR1
LI R2, TEXTEX
LI R3, ADDEX
BL @AFFICH
BL @ENTER
EFA2 BL @CLEAR1
LI R2, TEXT16
LI R3, ADD16
BL @AFFICH
```

### \* saisie du nom

```
SETO @DRAP5
B @MODIF1
```

### \* autre effacement?

```
AUTEFF LI R2, TEXT17
LI R3, ADD14
BL @AFFICH
CLR @DRAP
CLR @DRAP2
CLR @DRAP4
```

```
ARR2 BL @ARRET
CB R4, @OUI
JEQ EFA2
CB R4, @NON
JNE ARR2
B @OPT
```

### \* routine marque "E"

```
EFA1 LI R0, TPAB+118
MOVB @E, R1
BLWP @VSBW
```

### \* reenregistrement de la fiche

```
SETO @DRAP4
B @AJENR4
*****
```

### \* MODIFIER UN GROUPE

\*\*\*\*\*

```
E8375 BYTE 0
```

```
DRAP6 DATA 0
```

```
NOGR DATA 0
```

```
TEXTRET TEXT -'<BACK> = retour '
```

```
ADDRET DATA -926,0
```

```
TEXTTEF TEXT -' effacer dernier groupe '
```

```
TEXT -'Appuyez sur la barre <ESPACE> '
```

```
TEXT -'pour effacer le dernier groupe. '
```

```
TEXT -'<ENTER> = ARRET '
```

```
TEXTTCO TEXT -' corriger un groupe '
```

```
TEXT -'Rentrez le numero du groupe '
```

```
TEXT -'que vous voulez corriger '
```

```
TEXT -'No du groupe '
```

```
TEXTAJ TEXT -' ajouter un groupe '
```

```
TEXT -'Rentrez le nom du nouveau groupe '
```

```
TEXTNOM TEXT -'Nom du groupe '
```

```

TEXT -'~~~~~ '
ADDEF DATA 48,285,325,663,0
ADDAJ DATA 48,282,663,703,0
ADDCO DATA 48,285,327,543,0
ADDNOM DATA 663,703,0
ADD13 DATA 403,443,483,523,563,603,643,683
DATA 723,763,803,843,0
TEXT19 TEXT -'PLUS DE GROUPE LIBRE ... <ENTER> '
EVEN
*****
MODGR BL @CLEAR1
LI R2,TEXTG1
LI R3,ADD13
BL @AFFICH
LI R2,TEXRET
LI R3,ADDRET
BL @AFFICH
INS1 LI R3,8
CLR @LGR
LI R2,11
LI R0,487

* inscrire les groupes
INSGR LI R1,GROUPE
A @LGR,R1
BLWP @VMBW
A @D40,R0
A @D11,@LGR
DEC R3
JNE INSGR
MOV @DRAP7,@DRAP7
JNE SAISNO
MOV @DRAP6,@DRAP6
JNE INS5
SETO @DRAP6

* orientation
CB @RTCLA,@DEUX
JEQ CORNOM
CB @RTCLA,@UN
JEQ AJNOM

* effacer groupe
LI R2,TEXTEF
LI R3,ADDEF
BL @AFFICH
STOP3 BL @ARRET
BL @BLOQ
JNE STOP3
CB R4,@BLANC
JEQ EFFGR
CB R4,@ENT
JEQ INS6
CB R4,@BACK
JNE STOP3
CLR @DRAP6
B @NOMGR
EFFGR DEC @NGR
MOV @NGR,R2
MOVB @TROIS,@RTCLA
MPY @D11,R2
LI R1,GROUPE
A R3,R1
MOV R1,R5
LI R2,11
LI R3,NET
EFFG MOVB *R3,*R1+

DEC R2
JNE EFFG
CLR @DRAP6
JMP INS1
INS6 CLR @DRAP6
B @ENREGO

* ajouter un nom de groupe
AJNOM MOVB @UN,@E8375
LI R2,TEXTAJ
LI R3,ADDAJ
BL @AFFICH
JMP SGROUP
INS5 B @INS2

* corriger un nom de groupe
CORNOM LI R2,TEXTCO
LI R3,ADDCO
BL @AFFICH
MOVB @DEUX,@E8375

* saisie du No de groupe
SAISNO SETO @DRAP4
SETO @DRAP2
CLR @NOGR
LI R2,1
LI R0,557
LI R7,NOGR+1
MOV R2,@LSAIS
BL @ACCEPT
CB @NOGR+1,@UN
JL SAISNO
LI R1,-48
A R1,@NOGR
C @NOGR,@NGR
JH SAISNO
MOV @DRAP7,@DRAP7
JNE INS4

* affichage et saisie groupe
LI R2,TEXNOM
LI R3,ADDNOM
BL @AFFICH
MOV @NOGR,R1
MPY @D11,R1 -> result. R2
LI R0,784
LI R1,GROUPE-11
A R2,R1
LI R2,11
BLWP @VMBW

MOV @D11,@LSAIS
CLR @DRAP2
MOV R1,R7
BL @ACCEPT
B @INS1
INS4 B @AFFIG1

* saisie du nouveau nom
SGROUP LI R2,8
C @NGR,R2
JHE TOUSUT
MOV @D11,@LSAIS
LI R3,GROUPE
MOV @NGR,R1
MPY @D11,R1
A R2,R3
MOV R3,R7
LI R0,784
CLR @DRAP2
SETO @DRAP4
BL @ACCEPT
LI R0,784
BLWP @VSBW
CB R1,@BLANC
JEQ SGROUP
INC @NGR
CLR @DRAP6
B @INS1

* autre modification?
INS2 LI R2,TEXT15
LI R3,ADD14
BL @AFFICH
ARR4 BL @ARRET
CB R4,@NON
JNE INS3
CLR @DRAP6
B @ENREGO
INS3 CB R4,@OUI
JNE ARR4
CLR @DRAP6
MOVB @E8375,@RTCLA
B @MODGR

* tous les groupes sont utilises
TOUSUT LI R2,TEXT19
LI R3,ADD14
BL @AFFICH
BL @ARRET
CLR @DRAP6
B @ENREGO

*****
* PROGRAMME D'AFFICHAGE
*****
DRAP7 DATA 0
DRAP8 DATA 0
DRAP9 DATA 0
TEXT20 TEXT -'AFFICHER UN NOM '
TEXT -'~~~~~ '
TEXT -'Rentrez le nom de la personne '
TEXT -'que vous voulez afficher. '
TEXT -'NOM Prenom : '
ADD20 DATA 251,291,365,407,520,0
TEXT21 TEXT -'AFFICHER UN GROUPE '
TEXT -'~~~~~ '
TEXT -'No du groupe '
TEXT -'a afficher : '

```

ADD21 DATA 248,288,503,543,0

TEXT22 TEXT -' effacement '

TEXT23 TEXT -'Enregistrement No Marque: '

TEXT -'~~~~~'

TEXT -'~~~~~'

TEXT -'xxxxxxxxx'

TEXT -'x ordre x '

TEXT -'x alpha x '

TEXT -'x 000 x '

TEXT -'xxxxxxxxx'

TEXT -'Dans groupe '

TEXT -'~~~~~'

ADD23 DATA

200,240,311,351,391,431,471,509,549,0

AFFECR DATA 322,402,482,562,568,642,722,727,802

TEXT24 TEXT -' UN AUTRE AFFICHAGE ? O/N '

TEXT25 TEXT '<ENTER> autre fiche <BACK> arre'

TEXT -'t '

TEXT26 TEXT -'FIN DU FICHIER ... <ENTER> '

ADD26 DATA 527,0

EVEN

\*\*\*\*\*

\* une fiche

\*\*\*\*\*

RECH1 BL @CLEAR1

LI R2,TEXT2X

LI R3,ADDEX

BL @AFFICH

BL @ENTER

RECH0 BL @CLEAR1

LI R2,TEXT20

LI R3,ADD20

BL @AFFICH

SETO @DRAP7

B @MODIF1

\* No,actif,inactif,effacement

RECH2 BL @CLEAR1

LI R2,TEXT23

LI R3,ADD23

BL @AFFICH

C @NUMENR,@TRI

JH RECH7

MOV @SECT,R4

JMP RECH8

RECH7 MOV @NUMENR,R4

RECH8 LI R2,220

LI R7,217

BL @DECI

MOV @NUMENR,R4

LI R2,436

LI R7,433

BL @DECI

MOVB @BUFGR,R4

CB R4,@E

JNE ACT

LI R2,TEXT22

LI R3,ADDI

BL @AFFICH

JMP RECH3

ACT CB R4,@A

JNE INAC

LI R2,TEXTA

LI R3,ADDI

BL @AFFICH

JMP RECH3

INAC LI R2,TEXTI

LI R3,ADDI

BL @AFFICH

\* affichage informations

RECH3 LI R3,NBSAIS

LI R4,AFFECR

LI R5,TAMPON

RECH4 MOV \*R3+,R2

JEQ RECH5

MOV R5,R7

MOV \*R4+,R0

CLR R6

RE4 MOVB \*R7+,R1

CB R1,@POINT

JEQ RE5

BLWP @VSBW

INC R0

INC R6

C R6,R2

JL RE4

RE5 A R2,R5

JMP RECH4

\* affichage des groupes

RECH5 LI R3,BUFGR

LI R2,11

CLR @NLIG

CLR @LGR

MOV @NGR,R4

INC R4

RE1 DEC R4

JEQ RECH6

INC R3

LI R0,589

LI R1,GROUPE-11

A @D11,@LGR

CB \*R3,@OUI

JNE RE1

A @NLIG,R0

A @LGR,R1

BLWP @VMBW

A @D40,@NLIG

JMP RE1

\* autre fiche?

RECH6 MOV @DRAP8,@DRAP8

JNE SUITFI

LI R2,TEXT24

LI R3,ADDRET

BL @AFFICH

ARR5 BL @ARRET

CB R4,@NON

JNE RE2

B @OPT

RE2 CB R4,@OUI

JNE ARR5

CLR @DRAP

CLR @DRAP2

B @RECH0

\*\*\*\*\*

\* tout le fichier

\*\*\*\*\*

TOUTFI SETO @DRAP7

SETO @DRAP8

CLR @NUMENR

\* cas de tri temporaire

C @TRI,@TOT

JEQ TOUT1

MOV @CHTRI,@CHTRI

JEQ TOUT1

MOV @TRI,@LSAIS

MOV @TOT,@TRI

\* saisie de la fiche

TOUT1 INC @NUMENR

C @NUMENR,@TOT

JH FITERM

BL @PASS

BL @INPUT

BL @DSR

LI R0,TPAB

LI R1,TAMPON

LI R2,127

BLWP @VMBR

\* routine de recherche de groupe

MOV @DRAP9,@DRAP9

JEQ TOUT2

MOV @NOGR,R2

CB @BUFGR(R2),@OUI

JNE TOUT1

TOUT2 CLR @DRAP2

B @RECH2

\* <enter> ou <back>

SUITFI LI R2,TEXT25

LI R3,ADD14

BL @AFFICH

BL @ENTER

MOV @DRAP2,@DRAP2

JNE TOUT1

MOV @CHTRI,@CHTRI

JEQ FITER1

MOV @LSAIS,@TRI

B @OPT

\* fin du fichier

FITERM BL @CLEAR1

LI R2,TEXT26

LI R3,ADD26

BL @AFFICH

BL @ARRET

MOV @CHTRI,@CHTRI

JEQ FITER1

MOV @LSAIS,@TRI

FITER1 B @OPT

\*\*\*\*\*

\* un groupe

\*\*\*\*\*

AFFIGR BL @CLEAR1

LI R2,TEXT21

LI R3,ADD21

BL @AFFICH

SETO @DRAP6

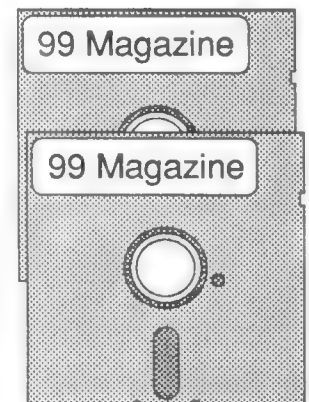
SETO @DRAP7

B @MODGR+4

\* recherche des fiches

AFFIG1 SETO @DRAP9

B @TOUTFI



\* s/p passage eventuel "index"

PASS C @NUMENR,@TRI

JH TOUT3

LI R1,INDEX

MOV @NUMENR,R2

DEC R2

SLA R2,1

A R2,R1

MOV \*R1,@SECT

JMP TOUT4

TOUT3 LI R1,NUMENR

TOUT4 LI R0,PAB+6

LI R2,2

BLWP @VMBW

RT

# Ticrok

Georges Goument

**J**e suis Ticrok le rouge. Si vous arrivez à me capturer, étant impulsif et coléreux de nature, je me détruirais en vous laissant le gain de la partie.

Ne rêvez pas trop quand même ; dans ma petite tête de monstre, j'ai plus d'un piège en réserve. Soyez assuré que, le moment venu, je saurais les utiliser contre vous.

Monstre je suis, mais bête... sûrement pas ! Bonne chance.

## Comment me capturer ?

Après chacun de vos déplacements, vous laissez une barrière sur place. Les barrières, je ne peux les franchir, tout comme vous d'ailleurs, alors attention ; ne vous piègez pas vous-même.

Vous devrez me bloquer étroitement pour que j'explose de rage.

De temps en temps, je suis un peu farceur et, pour vous narguer, je m'isole moi-même. Je peux me promener tranquillement, hors d'atteinte. Dans ce cas, vous perdrez également la partie.

A vous de trouver la parade pour

éviter qu'une telle situation se reproduise trop souvent.

Je suis quand même un chouette monstre pour vous révéler un de mes trucs. Malheureusement (pour vous), il y en a d'autres.

Vos déplacements s'effectuent à l'aide des touches 1 à 8. Le dessin situé sous le "terrain de jeu" vous indique les directions correspondantes.

Si vous me laissez m'isoler et que ma capture ne soit plus possible, appuyez sur "A".

Toutes les dix parties, le jeu s'arrête et affiche le résultat final.

99

## Programme Ticrok

### Basic TI

```
100 REM *****
110 REM *
120 REM * TICROK *
130 REM *
140 REM *****
150 REM *
160 REM * basic TI *
170 REM *
180 REM *****
190 REM *
200 REM * copyright *
210 REM *
220 REM *99 Magazine*
230 REM *
240 REM * G. Goument *
250 REM *
260 REM *****
270 PP=0
280 P1=0
290 P2=0
300 MY=1
310 MP=1
320 Z=0
330 CALL CLEAR
340 RANDOMIZE
350 CALL SCREEN(8)
360 FOR I=1 TO 13
370 CALL COLOR(I,1,1)
380 NEXT I
390 FOR I=40 TO 48
400 READ A$
```

```
410 CALL CHAR(I,A$)
420 NEXT I
430 DATA 991818FFFF181
899,00606018180606
01,001818181818189
9,000606181860608,
800000FEFE00008
440 DATA 8060601818060
6,99181818181818,0
106061818606,01000
07F7F000001
450 CALL CHAR(33,"1818
103854102844")
460 CALL CHAR(34,"3C7E
DBDB7E18A542")
470 CALL COLOR(1,13,1)
480 CALL CHAR(58,"0")
490 CALL CHAR(64,"FFFF
FFFFFFFFFFFF")
500 FOR I=72 TO 79
510 READ A$
520 CALL CHAR(I,A$)
530 NEXT I
540 DATA 1818103854102
844,03030922560910
2,0010D021320C1723
,020218162068D0C8,
181810315F,C4C8305
088422
550 DATA 10098A5E20E0D
8,C0C03C64A7
560 CALL CHAR(80,"3C7E
DBDB7E18A542")
570 CALL CHAR(88,"0000
0000AAFEAAAA")
```

```
580 CALL CHAR(120,"900
250008014400A")
590 PRINT TAB(12);"8:1
:2":TAB(12);":)*+:
":TAB(12);"70(,3":
TAB(12);":/.-:"TA
B(12);"6:5:4":::
600 CALL HCHAR(1,3,72)
610 CALL HCHAR(1,30,80
)
620 CALL HCHAR(1,6,58,
21)
630 CALL HCHAR(17,6,58
,8)
640 CALL HCHAR(17,19,5
8,8)
650 CALL VCHAR(2,6,58,
15)
660 CALL VCHAR(2,26,58
,15)
670 FOR I=7 TO 25
680 CALL VCHAR(2,I,64,
15)
690 NEXT I
700 FOR I=9 TO 11
710 CALL COLOR(I,6,1)
720 NEXT I
730 V=0
740 W=0
750 V=INT((11-2+1)*RND
)+2
760 W=INT((20-12+1)*RN
D)+12
770 CALL HCHAR(V,W,72)
780 VA=V
```

790 WA=W	1260 IF VA=2 THEN 1780	1730 IF A<>64 THEN 1780
800 Y=0	1270 V=VA-1	0
810 X=0	1280 W=WA	1740 CALL HCHAR(VA,WA,88)
820 Y=INT((11-2+1)*RND)+2	1290 GOTO 1720	1750 CALL SOUND(50,-7,0)
830 X=INT((20-12+1)*RND)+12	1300 IF T<>50 THEN 1360	1760 CALL HCHAR(V,W,72)
840 CALL GCHAR(Y,X,A)	1310 IF VA=2 THEN 1780	1770 GOTO 1940
850 IF A<>64 THEN 820	1320 IF WA=25 THEN 1780	1780 FOR I=1 TO 5
860 YA=Y	1330 V=VA-1	1790 IF A<>120 THEN 1840
870 XA=X	1340 W=WA+1	1800 CALL COLOR(12,10,15)
880 CALL HCHAR(Y,X,80)	1350 GOTO 1720	1810 CALL SOUND(10,-3,5)
890 CALL COLOR(5,15,15)	1360 IF T<>51 THEN 1410	1820 CALL COLOR(12,5,15)
900 CALL COLOR(6,14,15)	1370 IF WA=25 THEN 1780	1830 GOTO 1920
910 CALL COLOR(7,9,15)	1380 W=WA+1	1840 IF A=88 THEN 1890
920 CALL COLOR(8,7,15)	1390 V=VA	1850 CALL COLOR(7,13,15)
930 CALL COLOR(12,5,15)	1400 GOTO 1720	1860 CALL SOUND(10,-1,10)
940 CALL COLOR(2,16,13)	1410 IF T<>52 THEN 1470	1870 CALL COLOR(7,9,15)
950 CALL COLOR(3,16,13)	1420 IF WA=25 THEN 1780	1880 GOTO 1920
960 CALL COLOR(4,16,13)	1430 IF VA=16 THEN 1780	1890 CALL COLOR(8,10,15)
970 PP=PP+1	1440 V=VA+1	1900 CALL SOUND(10,-3,5)
980 TJ=0	1450 W=WA+1	1910 CALL COLOR(8,7,15)
990 N=0	1460 GOTO 1720	1920 NEXT I
1000 OB=0	1470 IF T<>53 THEN 1520	1930 GOTO 1230
1010 ND=0	1480 IF VA=16 THEN 1780	1940 IF O<DO THEN 2130
1020 O=0	1490 V=VA+1	1950 IF YA<=H THEN 1980
1030 DO=INT((10-2+1)*RND)+2	1500 W=WA	1960 IF YA>=B THEN 2010
1040 O=O+1	1510 GOTO 1720	1970 GOTO 2040
1050 IF O=DO*2 THEN 1020	1520 IF T<>54 THEN 1580	1980 IF XA<=G THEN 2680
1060 H=0	1530 IF VA=16 THEN 1780	1990 IF XA>=D THEN 2590
1070 B=0	1540 IF WA=7 THEN 1780	2000 GOTO 2410
1080 G=0	1550 V=VA+1	2010 IF XA<=G THEN 2860
1090 D=0	1560 W=WA-1	2020 IF XA>=D THEN 2770
1100 IF O<DO THEN 1150	1570 GOTO 1720	2030 GOTO 2500
1110 H=INT((9-3+1)*RND)+3	1580 IF T<>55 THEN 1630	2040 IF XA<=G THEN 2070
1120 B=INT((17-11+1)*RND)+11	1590 IF WA=7 THEN 1780	2050 IF XA>=D THEN 2100
1130 G=INT((13-7+1)*RND)+7	1600 W=WA-1	2060 GOTO 2130
1140 D=INT((25-19+1)*RND)+19	1610 V=VA	2070 IF YA<=H THEN 2680
1150 XA=X	1620 GOTO 1720	2080 IF YA>=B THEN 2860
1160 YA=Y	1630 IF T<>56 THEN 1690	2090 GOTO 2230
1170 VA=V	1640 IF WA=7 THEN 1780	2100 IF YA<=H THEN 2590
1180 WA=W	1650 IF VA=2 THEN 1780	
1190 V=0	1660 V=VA-1	
1200 W=0	1670 W=WA-1	
1210 Y=0	1680 GOTO 1720	
1220 X=0	1690 IF T<>65 THEN 1230	
1230 CALL KEY(3,T,S)	1700 CALL HCHAR(VA,WA,64)	
1240 IF S=0 THEN 1230	1710 GOTO 4410	
1250 IF T<>49 THEN 1300	1720 CALL GCHAR(V,W,A)	



0	2680 GOSUB 3210	3190 X=X+1
2110 IF YA>=B THEN 277	2690 GOSUB 3130	3200 GOTO 3650
0	2700 GOSUB 3300	3210 IF YA=16 THEN 303
2120 GOTO 2320	2710 GOSUB 3380	0
2130 IF V<=YA THEN 217	2720 GOSUB 3040	3220 IF XA=25 THEN 303
0	2730 GOSUB 2950	0
2140 IF W<=XA THEN 259	2740 GOSUB 3470	3230 CALL GCHAR(YA+1,X
0	2750 GOSUB 3550	A+1,U)
2150 IF W>XA THEN 2680	2760 GOTO 4110	3240 IF U<>120 THEN 32
2160 GOTO 2410	2770 GOSUB 3550	70
2170 IF V=YA THEN 2210	2780 GOSUB 3470	3250 Z=1
2180 IF W<XA THEN 2770	2790 GOSUB 2950	3260 GOTO 3280
2190 IF W>=XA THEN 286	2800 GOSUB 3380	3270 IF U<>64 THEN 303
0	2810 GOSUB 3040	0
2200 GOTO 2500	2820 GOSUB 3130	3280 X=X+1
2210 IF W<XA THEN 2320	2830 GOSUB 3300	3290 GOTO 3670
2220 IF W>XA THEN 2230	2840 GOSUB 3210	3300 IF YA=16 THEN 303
2230 GOSUB 3130	2850 GOTO 4110	0
2240 GOSUB 3040	2860 GOSUB 3040	3310 CALL GCHAR(YA+1,X
2250 GOSUB 3210	2870 GOSUB 2950	A,U)
2260 GOSUB 2950	2880 GOSUB 3130	3320 IF U<>120 THEN 33
2270 GOSUB 3300	2890 GOSUB 3550	50
2280 GOSUB 3550	2900 GOSUB 3210	3330 Z=1
2290 GOSUB 3380	2910 GOSUB 3300	3340 GOTO 3360
2300 GOSUB 3470	2920 GOSUB 3470	3350 IF U<>64 THEN 303
2310 GOTO 4110	2930 GOSUB 3380	0
2320 GOSUB 3470	2940 GOTO 4110	3360 Y=Y+1
2330 GOSUB 3380	2950 IF YA=2 THEN 3030	3370 GOTO 3690
2340 GOSUB 3550	2960 CALL GCHAR(YA-1,X	3380 IF YA=16 THEN 303
2350 GOSUB 3300	A,U)	0
2360 GOSUB 2950	2970 IF U<>120 THEN 30	3390 IF XA=7 THEN 3030
2370 GOSUB 3210	00	3400 CALL GCHAR(YA+1,X
2380 GOSUB 3040	2980 Z=1	A-1,U)
2390 GOSUB 3130	2990 GOTO 3010	3410 IF U<>120 THEN 34
2400 GOTO 4110	3000 IF U<>64 THEN 303	40
2410 GOSUB 3300	0	3420 Z=1
2420 GOSUB 3380	3010 Y=Y-1	3430 GOTO 3450
2430 GOSUB 3210	3020 GOTO 3690	3440 IF U<>64 THEN 303
2440 GOSUB 3470	3030 RETURN	0
2450 GOSUB 3130	3040 IF YA=2 THEN 3030	3450 X=X-1
2460 GOSUB 3550	3050 IF XA=25 THEN 303	3460 GOTO 3670
2470 GOSUB 3040	0	3470 IF XA=7 THEN 3030
2480 GOSUB 2950	3060 CALL GCHAR(YA-1,X	3480 CALL GCHAR(YA,XA-
2490 GOTO 4110	A+1,U)	1,U)
2500 GOSUB 2950	3070 IF U<>120 THEN 31	3490 IF U<>120 THEN 35
2510 GOSUB 3040	00	20
2520 GOSUB 3550	3080 Z=1	3500 Z=1
2530 GOSUB 3130	3090 GOTO 3110	3510 GOTO 3530
2540 GOSUB 3470	3100 IF U<>64 THEN 303	3520 IF U<>64 THEN 303
2550 GOSUB 3210	0	0
2560 GOSUB 3380	3110 X=X+1	3530 X=X-1
2570 GOSUB 3300	3120 GOTO 3630	3540 GOTO 3650
2580 GOTO 4110	3130 IF XA=25 THEN 303	3550 IF YA=2 THEN 3030
2590 GOSUB 3380	0	3560 IF XA=7 THEN 3030
2600 GOSUB 3470	3140 CALL GCHAR(YA,XA+	3570 CALL GCHAR(YA-1,X
2610 GOSUB 3300	1,U)	A-1,U)
2620 GOSUB 3550	3150 IF U<>120 THEN 31	3580 IF U<>120 THEN 36
2630 GOSUB 3210	80	10
2640 GOSUB 2950	3160 Z=1	3590 Z=1
2650 GOSUB 3130	3170 GOTO 3190	3600 GOTO 3620
2660 GOSUB 3040	3180 IF U<>64 THEN 303	3610 IF U<>64 THEN 303
2670 GOTO 4110	0	0

3620 X=XA-1	4100 GOTO 4230	4530 VA=V
3630 Y=YA-1	4110 E=1	4540 CALL HCHAR(VA,10,32)
3640 GOTO 3700	4120 P2=P2+1	4550 V=V+1
3650 Y=YA	4130 CALL HCHAR(YA,XA,32)	4560 A=A+1
3660 GOTO 3700	4140 FOR P=0 TO 30 STEP 3	4570 NEXT P
3670 Y=YA+1	4150 CALL SCREEN(E)	4580 IF WA>16 THEN 4610
3680 GOTO 3700	4160 CALL SOUND(50,-5,P)	4590 CALL HCHAR(24,10,76)
3690 X=XA	4170 E=E+1	4600 GOTO 4630
3700 IF ND>0 THEN 3750	4180 NEXT P	4610 CALL HCHAR(24,20,79)
3710 IF P2>P1 THEN 3740	4190 CALL SCREEN(8)	4620 VA=0
3720 ND=INT((25-15+1)*RND)+10	4200 MY=MY+2	4630 V=0
3730 GOTO 3750	4210 CALL HCHAR(MY,3,34)	4640 A=0
3740 ND=INT((15-5+1)*RND)+5	4220 GOTO 4780	4650 CALL SCREEN(8)
3750 TJ=TJ+1	4230 CALL GCHAR(V-1,W,U)	4660 MP=MP+2
3760 IF TJ<ND THEN 3970	4240 IF U=64 THEN 1040	4670 CALL HCHAR(MP,30,33)
3770 TJ=0	4250 CALL GCHAR(V-1,W+1,U)	4680 P1=P1+1
3780 ND=0	4260 IF U=64 THEN 1040	4690 GOTO 4780
3790 DM=0	4270 CALL GCHAR(V,W+1,U)	4700 FOR I=1 TO LEN(M\$)
3800 OB=INT((6-1+1)*RND)+1	4280 IF U=64 THEN 1040	4710 CALL HCHAR(22,2+I,ASC(SEG\$(M\$,I,1)))
3810 IF OB>2 THEN 3870	4290 CALL GCHAR(V+1,W+1,U)	4720 CALL SOUND(1,-1,20)
3820 IF P1>=P2 THEN 3850	4300 IF U=64 THEN 1040	4730 NEXT I
3830 OB=8	4310 CALL GCHAR(V+1,W,U)	4740 RETURN
3840 GOTO 3970	4320 IF U=64 THEN 1040	4750 CALL HCHAR(22,3,32,28)
3850 OB=4	4330 CALL GCHAR(V+1,W-1,U)	4760 M\$=""
3860 GOTO 3970	4340 IF U=64 THEN 1040	4770 RETURN
3870 IF OB>4 THEN 3930	4350 CALL GCHAR(V,W-1,U)	4780 IF PP=10 THEN 4900
3880 IF P1>=P2 THEN 3910	4360 IF U=64 THEN 1040	4790 M\$="nouvelle partie o n"
3890 OB=9	4370 CALL GCHAR(V-1,W-1,U)	4800 GOSUB 4700
3900 GOTO 3970	4380 IF U=64 THEN 1040	4810 CALL KEY(3,T,S)
3910 OB=5	4390 CALL HCHAR(V,W,32)	4820 IF S=0 THEN 4810
3920 GOTO 3970	4400 WA=W	4830 IF T<>78 THEN 4850
3930 IF P1>=P2 THEN 3960	4410 A=73	4840 GOTO 4900
3940 OB=10	4420 V=18	4850 IF T<>79 THEN 4810
3950 GOTO 3970	4430 FOR P=1000 TO 400 STEP -100	4860 GOSUB 4750
3960 OB=6	4440 CALL SCREEN(16)	4870 CALL HCHAR(24,10,32)
3970 IF OB=0 THEN 4030	4450 CALL COLOR(6,2,1)	4880 CALL HCHAR(24,20,32)
3980 CALL HCHAR(YA,XA,120)	4460 CALL SOUND(1,P,10)	4890 GOTO 670
3990 DP=DP+1	4470 IF WA<=16 THEN 4520	4900 CALL CLEAR
4000 IF DP<OB THEN 4080	4480 CALL HCHAR(V,20,A)	4910 PRINT "nombre de parties jouees ";PP
4010 OB=0	4490 VA=V	4920 PRINT ":::::"
4020 DP=0	4500 CALL HCHAR(VA,20,32)	4930 PRINT "pour vous ";P2
4030 IF Z=0 THEN 4070	4510 GOTO 4550	4940 END
4040 CALL HCHAR(YA,XA,120)	4520 CALL HCHAR(V,10,A)	
4050 Z=0		
4060 GOTO 4080		
4070 CALL HCHAR(YA,XA,64)		
4080 CALL SOUND(50,-5,0)		
4090 CALL HCHAR(Y,X,80)		

# Un nouveau module graphique pour le TI-99/4A

Henri Mathian

**U**n nouveau module conçu et fabriqué par MECHATRONIC, de Sindelfingen en Allemagne, est disponible en France depuis début juin. Son nom : "Le Basic Graphique Etendu Apesoft". Ses capacités : "Super !". Il apporte une nouvelle jeunesse à notre bon vieux TEXAS avec la quasi-totalité des commandes graphiques disponibles sur le micro-pomme. De plus il est accompagné d'un manuel en français. Tout pour plaire ce petit module... même son prix (1200,00 F à la Règle à Calcul !).

Cet article n'a pour but que de donner un rapide aperçu des possibilités de ce module qui possède 44 Ko de mémoire préprogrammée.

Le principe est très simple : le déplacement d'un curseur graphique génère le tracé d'une ligne. L'écran sert de feuille de travail. A l'initialisation du "mode graphique" une fenêtre est définie à l'intérieur de laquelle le curseur partant d'un point donné va, selon les instructions reçues, tracer la courbe demandée avec une grande rapidité.

L'adressage de chaque point de la fenêtre graphique devient possible, et cela en 16 couleurs, pour le premier et l'arrière plan. 40 instructions graphiques sont disponibles et peuvent être appelées sous le TI-Basic ou le TI-Basic Etendu. La composition

de programmes facilement agrémentée de graphiques haute résolution devient possible. Contrairement à ce que dit la brochure en page 42, les sous-programmes CALL SOUND et CALL SPRITE peuvent être utilisés même en mode graphique. Ainsi il est possible d'ouvrir une fenêtre, de créer un graphique, de le dupliquer éventuellement sur une autre partie de l'écran ou de le faire se déplacer, de définir un ou plusieurs lutins, de les appeler, de les déplacer sur fond musical sans altérer le graphisme préalablement composé.

Ce module comporte aussi une instruction très pratique "CALL BHCOPY" qui permet de réaliser une copie d'écran en mode Bit-Map sur imprimantes à aiguilles de types GP 550, EPSON FX-80, RX-80 et compatibles. Avec toutefois une petite restriction : Les SPRITES ne sont pas imprimés.

L'utilisation de ce module nécessite obligatoirement la présence de l'extension 32 Ko.

"C'est extraordinaire de voir fonctionner le XG Basic Apesoft" dit la brochure. "Ce n'est pas faux !" répond l'utilisateur.

## Mode de fonctionnement

Le déplacement d'un curseur génère le tracé d'une ligne à

l'intérieur d'une fenêtre graphique qui comporte 128 colonnes (16 caractères \* 8 points) et 120 lignes (15 caractères \* 8 points) soit 15 360 points.

Cette fenêtre peut être affichée seule en un endroit quelconque de l'écran, en partie ou totalement en dehors de l'écran, elle peut être reproduite une ou plusieurs fois sur la totalité de l'écran en juxtaposition, superposition partielle, être déplacée à la manière d'un escargot (boucle FOR-NEXT avec effacement de la fenêtre précédente), être découpée ou être réduite (affichage d'une portion de la fenêtre).

La figure exécutée dans la fenêtre graphique est unique. On ne peut ouvrir deux fenêtres à l'écran et y tracer des figures différentes. Le nouveau tracé destiné à la fenêtre 3 ira simultanément s'inscrire en superposition dans les 3 fenêtres ouvertes. C'est dommage ! A nous de trouver le truc !

La limitation à 15 360 points est nécessaire afin de réserver suffisamment de place dans la VDP-RAM pour les programmes Basic, les variables alpha-numériques, etc...

L'adressage direct de la totalité de l'écran soit 256 colonnes (32 caractères \* 8 points) et 196 lignes (24 caractères \* 8 points), ce qui représente 50 176 points, demande 12 Ko de la

VDP-RAM ; les tableaux de caractères et de couleurs débordent sur les tables d'adresses de l'interpréteur Basic. Si l'adressage individuel de ces 50 176 points est possible, la fenêtre ouverte à l'écran pour créer le graphique ne peut malheureusement dépasser 15 360 points dans la version actuelle du logiciel.

## Quelques instructions graphiques

### CALL APESOFTE

Le sous-programme APESOFTE ferme tous les fichiers encore ouverts, transfère les routines graphiques depuis le module vers l'extension mémoire 32 Ko, réserve la place nécessaire pour le graphique dans la VDP-RAM puis exécute une instruction NEW de façon à effacer tous les programmes se trouvant en mémoire.

Les routines graphiques sont constituées d'instructions très puissantes qui simplifient énormément la réalisation de dessins complexes.

### CALL LINK("GRAFIC", MODUS)

Annonce au calculateur le mode graphique et initialise tous les registres.

### CALL LINK("WINDOW", Colonne,Ligne)

Positionne la fenêtre graphique à l'écran.

### CALL LINK("WINDOW",Z,S, ZA,SA,DZ,DS)

Positionne une ou plusieurs portions de la fenêtre.

### CALL LINK("SETBLE", Nombre de colonnes)

Permet d'obtenir une fenêtre à dimensions variables (la surface maximale restant toujours, dans l'état actuel de mes connaissances, de 15 360 pts).

### CALL LINK("SETCOL",

### Couleur Premier Plan, Couleur Arrière Plan)

Chacune des 16 couleurs du Basic TI peut être utilisée soit dans le premier plan, soit dans l'arrière plan. On peut utiliser simultanément dans un graphique plusieurs couleurs au premier et à l'arrière plans.

### CALL LINK("INVERT",X,Y, DX,DY)

Intervient les couleurs du premier et de l'arrière plans.

### CALL LINK("CLSCRN")

Equivalent à CALL CLEAR, efface le graphique, les paramètres internes du curseur restant inchangés.

### CALL LINK("CENTRE",X,Y)

Définit le système de coordonnées utilisateur.

### CALL LINK("MOVE", Distance)

Permet de tracer une ligne de longueur "Distance" à partir de la position actuelle X,Y du curseur.

### CALL LINK("MOVETO",X,Y)

Trace une ligne depuis la position actuelle du curseur jusqu'au point défini par les coordonnées X et Y.

### CALL LINK("TURN",PHI)

Ajoute la valeur PHI exprimée en degrés à l'angle courant du curseur.

### CALL LINK("RECT",A,B)

Trace un rectangle.

### CALL LINK("CIRCLE",X,Y, R)

Trace un cercle de centre(X,Y) de rayon R.

### CALL LINK("ARCUS",X,Y,R, PHI,DPHI)

Trace un arc circulaire.

### CALL LINK("ELLIPS",X,Y, A,B)

Trace une ellipse centrée en X,Y de demi-grand axe A, de demi-

petit axe B.

### CALL LINK("VALUES",X,Y, PHI,FG,BG)

Affecte aux variables désignées les valeurs actuelles des paramètres. Coordonnées curseur (X,Y). Angle curseur (PHI). Couleur premier plan (FG). Couleur de l'arrière plan.

### CALL LINK("HSTDIA",X,Y, WIDTH,HEIGHT,DEPTH)

Trace une barre de diagramme.

## Composition graphique

### CALL LINK("WRITE",Z,S, STRING)

Permet le mélange de graphique et de texte.

### CALL LINK("DSPLAY",Z,S, SIZE,VAR\$)

Correspond à "DISPLAY AT".

### CALL LINK("ACCEPT",Z,S, SIZE,VAR\$)

Correspond à "ACCEPT AT"

## Sauvegarde

### CALL GSAVE("DATEINAME")

Les dessins ne peuvent être stockés que sur floppy disques.

### CALL GLOAD("DATEINAME")

Cette instruction charge dans le VDP-RAM le dessin enregistré sur la disquette. L'appel de GLOAD sans avoir, au préalable, appelé "GRAFIC", entraîne un message d'erreur.

## Quelques instructions non graphiques

qui viennent s'ajouter à l'ensemble des commandes et instructions du TI Extended Basic que possède également ce module :

### CALL BHCOPY("FILENAME", "ESC-SEQUENZ")

Réalise une copie d'écran. ex.:

**CALL BHCOPY("PIO","K")**  
produit une copie "parallèle" sur une BMC.

#### **CALL VPEEK**

Lecture des positions de mémoire dans la VDP-RAM.

#### **CALL VPOKE**

Ecriture directe dans les positions de la VDP-RAM.

#### **CALL GPEEK**

Retourne les caractères contenus dans les valeurs ASCII 32 à 126 à leurs définition de base. Très proche de CALL CHARSET.

#### **CALL WAIT**

Attente de 0 à 16 382 (1000 correspond à 20 secondes).

#### **CALL MOVE**

Déplace le contenu de blocs de mémoire dans l'ordinateur.

#### **CALL MSAVE**

Enregistrement sur cassette de programmes en langage machine.

#### **CALL MLOAD**

Charge le contenu de la mémoire vive, sauvegardé par MSAVE.

#### **CALL BYE**

Peut être utilisé en cours de programme.

#### **CALL NEW**

Peut être utilisé à l'intérieur d'un programme.

#### **CALL RESTORE (A)**

Instruction RESTORE comportant une variable.

#### **CALL QUIT OF**

Met la touche QUIT hors fonction.

#### **CALL QUIT ON**

Rétablit la fonction.

#### **CALL SPRON**

Mise en mouvement de tous les sprites au même moment.

#### **CALL SPROF**

Arrête le mouvement de tous les sprites.

#### **CALL SCREENON**

Allume l'écran.

#### **CALL SCREENOF**

Eteint l'écran.

#### **CALL FIND("Cherchestring", "Stringarry"(),Variable retour)**

Dans un alignement de Strings d'une seule dimension, le sous-programme FIND cherche la notion définie dans "Cherchestrings". La variable retour numérique prend la valeur de la notion recherchée dans l'alignement.

Cette liste d'instructions graphiques ou non graphiques n'est pas exhaustive. Elle donne une idée des possibilités surprenantes de ce petit module qui donne vraiment envie de continuer, ou de se remettre, à tapoter les touches de ce cher vieux TEXAS.

**99**

## ***Le module "MAXIMEM 48Ko" pour TI-99/4A et TI-99/4***

Ce nouveau module, qui nous a été présenté sous la forme d'un prototype, offre de nombreuses possibilités :

- il double la capacité mémoire du TI 99-4 ou 4/A ;
- Maximem se connecte à l'endroit réservé aux modules ;
- il ne nécessite aucune alimentation extérieure ;
- il permet de charger et exécuter automatiquement tout module mis préalablement sur disque ;
- Maximem peut rester indéfiniment en place, ce qui évite l'usure du port d'entrée de la console ;
- Maximem met à votre disposition plusieurs choix de modules après la mire de couleurs, par exemple : Editeur/Assembleur, Basic Etendu, Zero Zap, Basic TI.

### **Utilisation**

Maximem est simple à utiliser : après la mire de couleurs, trois choix s'offrent à l'utilisateur :

- 1) TI Basic (rien n'est changé pour le moment) ;
- 2) MAXIMEM (qui charge le module de votre choix et lance son exécution) ;
- 3) Editeur/assembleur (une version améliorée du module Editor/assembleur).

### **Caractéristiques techniques**

Capacité 48Ko répartis comme suit (toutes les adresses en hexadécimal) :

- 6000 à 7FFF RAM 1 (8Ko)
- 6000 à 7FFF RAM 2 (8Ko)  
(les RAM 1 et RAM 2 sont commutables par "flag")
- 6000 à 77FF GRAM (6Ko)

- 8000 à 97FF GRAM (6Ko)
- A000 à B7FF GRAM (6Ko)
- C000 à D7FF GRAM (6Ko)
- E000 à FFFF GRAM (8Ko)

### **Equipement requis**

L'utilisation de ce module requiert un lecteur de disquette avec son contrôleur et une extension de mémoire 32Ko.

A l'avenir, une version cassette de MAXIMEM permettra l'usage seul du magnétophone à cassettes pour les modules ne nécessitant pas l'extension mémoire.

Nous ne savons pas encore quel sera le prix de ce très intéressant module, ni par qui il sera distribué. Si vous désirez obtenir des précisions, vous pouvez contacter le fabricant :

*Gournay Guy  
933, Delorimier  
Longueuil  
J4K 3M8  
PQ Canada*

**99**



# Courrier des Lecteurs

Alexandre Duback

Je viens d'acheter la Mini-mémoire et j'ai réussi à subtiliser le livre "Initiation au langage assembleur du TI-99/4A", de Denise Amrouche et Roger Didi pour l'étudier et voilà mon problème : je suis au chapitre 5, étude de la VDP RAM.

Il est dit (page 38), à propos des lutins, que la table des descriptions de leur forme s'étend de >0400 à >07FF ; soit 1024 octets. Il y a possibilité de définir 128 lutins. On nous dit aussi (page 39), à propos des vitesses, que la table s'étend de >0780 à >07FF. Cette table des vitesses empiète donc sur les 1024 octets réservés à la définition des 128 lutins. Pourquoi ? Si ces adresses sont justes, cela veut dire que je ne peux définir que 112 lutins : >0400 à >077F (page 40).

Il y a aussi un autre problème : il est possible de définir 256 caractères, moins les trente deux premiers : >0800 à >0FFF. Nous

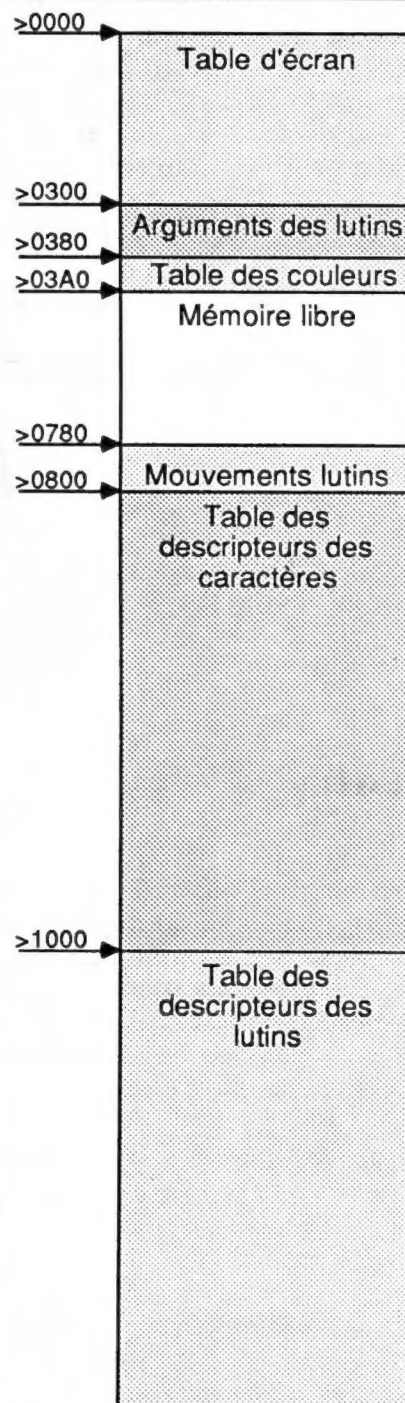
pouvons facilement trouver leur code ASCII. Quand je définis un lutin, je lui donne un code ASCII allant généralement de 128 à 256 (Si il y en a 128 définissables). Je peut aussi définir des caractères allant de 128 à 256. Quand je demande l'apparition d'un cube à l'écran, de code ASCII 128, que celui-ci est un lutin défini dans la table des lutins et aussi défini dans la table des caractères, est-ce la forme du lutin ou celle du caractère qui va apparaître sur l'écran, ayant tous deux le même code ASCII ?

Frédéric Rejol - 42100 St Etienne

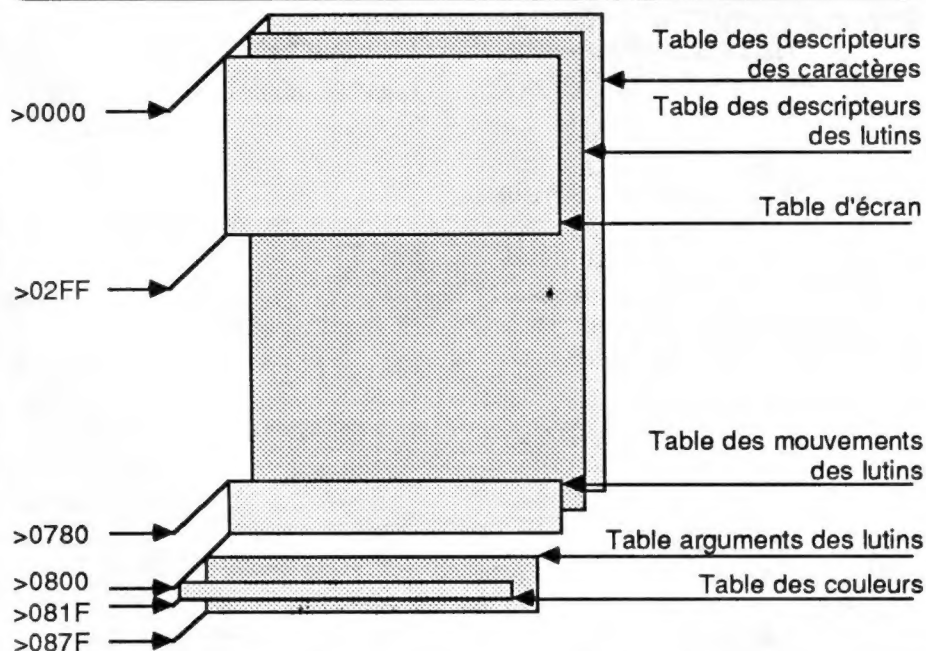
Avec la Mini-mémoire et le Basic TI, les différentes tables utilisées pour l'affichage des caractères et lutins se superposent, ceci afin de laisser de la place pour le programme Basic qui est, lui aussi, loger dans la mémoire vidéo (seule zone de mémoire vive disponible sans extension 32Ko, si l'on ne tient pas compte des 256 octets du TMS 9900). Le

schéma ci-joint visualise la mémoire et explique que les possibilités se trouvent réduites par cette configuration par défaut.

## Exemple de mémoire vidéo configurée depuis un programme en assembleur



## Configuration par défaut de la mémoire vidéo lors de l'emploi du Basic TI



Fort heureusement, en assembleur - puisque c'est ce qui semble vous préoccuper - il est parfaitement possible de partitionner la mémoire dans le but d'obtenir des tables complètes. Le second schéma proposé montre une configuration possible parmi d'autres. Puisque ceci n'est peut-être pas évident, quelques rappels et explications s'imposent.

L'adresse de base de la majorité des tables n'est pas figée puisqu'elle est fonction de l'état de l'un des registres du processeur vidéo. Ainsi, le contenu du registre 2 fixe l'adresse de base de la table d'écran, le registre 3 fixe l'adresse de base de la table des couleurs, etc... Seule la table utilisée pour les mouvements des lutins doit impérativement se trouver en >780 (1920).

Pour fixer l'adresse de base de la

table d'écran, il faut placer dans le registre 2 une valeur correspondante à l'adresse voulue divisée par >400 (1024). Dans notre exemple, l'adresse de base de la table d'écran est >0000, on place donc dans le registre 2 : >0000/>400, soit 0.

Pour la table des couleurs, même méthode mais il faut cette fois utiliser le registre 3, et diviser l'adresse de base désirée par >40 (64). Pour que l'adresse de base de la table des couleurs soit en >380 (896), il faut placer la valeur >E (14).

Le registre 4 est employé pour la table des descripteurs des caractères. La valeur à placer dans le registre doit correspondre à l'adresse de base voulue, divisée par >800 (2048). Dans le cas qui nous intéresse, et où nous avons situé la table en >800, le registre vidéo concerné doit contenir >800/>800, soit 1.

Pour la table des argument des

lutins (couleurs, positions, etc...) on utilise le registre 5, dans lequel on place l'adresse de base de la table, divisée par >80 (128). Dans le cas où l'adresse de base doit être >300 (768), on place 6 dans le registre 5.

Le registre 6 définit l'adresse de base de la table des descripteurs des lutins (formes). la valeur à indiquer correspond à l'adresse de base divisée par >800 (2048). Pour une adresse de base en >1000 (4096), il faut placer dans le registre 6 la valeur 2 ( $2 * 2048 = 4096$ ).

Cette méthode permet d'obtenir 256 caractères utilisables, avec 32 groupes de 8 caractères pour les couleurs, et 32 lutins avec 256 définitions différentes au même moment (ou 64 dans le cas de lutins en 16\*16 pixels).

99

## Petites annonces (gratuites)

### Ventes

A vendre : TI-99/4A bon état (4.83), Basic étendu, câble cassettes, Mini-mémoire, joysticks et huit modules de jeux (Parsec, Star Trek, Burger Time...) plus nombreux jeux sur cassettes et livres. Le tout : 3500 FF.

Gilles Bompard - 12, place Béranger - 77170 Servon - Tél. : 405.11.66.

Vends TI-99/4A, Basic étendu, Mini-mémoire, Echecs, livres et cassettes. Très bon état. Prix à débattre.

Roger Feyssaguet - Chizé - 79170 Brioux sur Boutonne - Tél. : 16 (49) 76.70.76 après 19H.

Vend, cause double emploi, boîtier d'extension périphériques avec câble et carte de commande : 700 FF ; extension mémoire 32Ko extérieure sous garantie : 1000 FF. Le lot : 1500 FF.

M. Lepercque - Tél. : (1) 563.12.12 poste 3748 (heures bureau).

Vends boîtier d'extension avec contrôleur et lecteur de disquettes, carte mémoire 32Ko et carte RS232 : 7000 FF. Editeur/assembleur : 500 FF. TI

Writer Word Processor : 800 FF. TI LOGO II : 800 FF.

Jean-Pierre IVES - 5, rue de l'Argile - 34160 Castries - Tél. : (67) 70.07.74.

Vends synthétiseur de paroles : 400 FF.

J-F. de Belleval - La Borne Trouée - 60153 Rethondes - Tél. : (4) 485.62.26 après 18H.

Vends TI-99/4A, câble magnéto, joysticks, Basic étendu, Parsec, TI Invaders, Lunar Jumper, Car Wars, Gestion de fichiers, livres, "99 Magazine" 1 à 7, plus de nombreux programmes. Valeur : plus de 3500 FF ; vendu 2500 FF.



Olivier Serreault - Plan de Leydet  
- 04200 Sisteron.

Vends nombreuses cassettes en Basic TI et Basic étendu ("99 Magazine" et "Hebdogiciel") pas cher et en bon état.

Pierre-E. Gougelet - 28, rue Guillaumet - 51400 Livry-Louvercy.

Vends TI-99/4A Péritel, Basic étendu Français, TI LOGO II, mémoire 32Ko externe, interface parallèle externe, magnéto TI, deux câbles cassettes, numéros 1 à 8 de "99 Magazine" avec cassettes d'accompagnement, modules Echecs, Gestion privée, Gestion de fichiers, manettes de jeux TI, cassette "Basic par soi-même", livres ("Boîte à outils", "A l'affiche", "102 programmes", "Jeux, trucs et comptes").

Serge Nardizzi - résidence Les Jardins de l'Aude - Batiment E - 11000 Carcassonne - Tél. : (68) 47.59.82 après 20H30.

Vends boîtier d'extension, carte mémoire 32Ko, carte RS232 (sorties série et parallèle), contrôleur et lecteur de disquettes, imprimante Star Gemini 10 (120 cps, avec cordon), synthétiseur vocal, modules Basic étendu, TI-Writer, TI LOGO 2, Gestion de fichiers, Gestion privée et TI Calc.

Michel Viaene - Longueville sue Aube - 10170 Méry sur Seine - Tél. : (25) 21.22.32.

Vends modules Tombstone, Video Games I, Wumpus, Star Trek, Mash, Car Wars, Chilshom Trail, A maze ing : 100 FF chacun. Cassettes Rétro I et II, Lunar Lander, Lunar Jumper, Solar System : 50 FF chaque.

J. Reibel - 9, square V. Fleming - 92350 Le Plessis Robinson - Tél. : 16 (1) 631.46.11.

Vends modules Basic étendu, Hunt the Wumpus, et les cassettes Lunar Jumper, Treasure Trap et Pilot. Tous avec manuels, vente ensemble ou séparément.

M. Heudes - 17, rue de

Léningrad - 75008 Paris - Tél. : 522.78.48.

Vends modules TI Invaders, Football, Blast, Munch Man : 120 FF pièce ; Pole Position, Parsec : 160 FF pièce. Vends "99 Magazine" numéros 1 à 8 : 20 FF le numéro. Le tout est en très bon état.

T. Martel - 12, rue Bonne Santé - 76620 Le Havre - Tél. : (35) 45.30.58.

Vends TI-99/4A, poignées de jeux, câble cassettes, deux livres, TI Invaders, Hustle, Star Trek, Music Maker, huit cassettes "99 Magazine". Valeur 2800 FF ; vendu 1400 FF.

P. Petitjean - 7, rue d'Alembert - 18000 Bourges - Tél. : (48) 65.62.54.

Vends TI-99/4A avec Basic étendu en Français, cassette "Basic par soi-même", autres cassettes et livres de programmes : 2000 FF.

J-M. Lesage - Pré Bercy III - 03000 Moulins - Tél. : (70) 20.13.84.

## Achats

Achète disquette "TI Writer".

G. Crittin - 14 bis, chemin du Bochet - 1196 Gland (Suisse) - Tél. : 19.41.22.64.38.50.

Achète contrôleur de disquettes seul 1500 FF ou avec lecteur de disquettes simple face 3500 FF (à débattre). Faire offre à :

Pascal Thomas - Paris - Tél. : (1) 203.71.10.

Achète modules TI Calc, Multiplan, Stat, Echecs, Maths (Collins), etc... Interface parallèle ou série pour imprimante et console, même en panne.

M. Voisin - 22, route de Vewerque - Grépiat - Tél. : (61) 08.21.88.

Achète module Mini-mémoire complet (manuel d'utilisation).

Jean-Louis Philippe - 73, route

de Sierck - Koenigsmaker - 57110 Yutz - Tél. : 16 (8) 250.01.46.

Cherche mode d'emploi de la Mini-mémoire, Modules TI Writer, Statistiques, Physical Fitness, Numération I & II, Magie des nombres, disquette Dictée magique, plans et schémas de l'interface RS232C (offre possibilité réalisations) et/ou extension mémoire 32Ko.

Philippe Mechelaere - 6, rue des Saints Sauveurs - 92260 Fontenay aux Roses - Tél. : 16 (1) 350.88.18 après 20H (ou laisser message).

Achète module "Speech Editor" (200 à 300 FF). Faire offre à :

M. Couegnât - 66, rue du Repas - 69007 Lyon - Tél. : 16 (1) 858.21.16.

Cherche carte interface RS232C pour boîtier d'extension, module TI Calc, modules et disquettes Multiplan et TI Writer, console TI-99/4A et accessoires hors service pour bricolages.

Henri Espi - résidence Le Poséidon - 66750 Saint Cyprien Plage - Tél. : (68) 21.22.12 après 19H.

Recherche Modules "Speech Editor" et "LOGO 2". Carte mémoire 32Ko, lecteur et contrôleur de disquettes internes. Etudie toutes propositions.

Guy Mahé - 23, allée H. Sellier - A.D 1156 - 92800 Puteaux.

## Divers

Echange manuel "Editor/Assembler". Faire offre à :

Gilbert Vigliano - Lot "La Plage" - Batiment A - avenue Jean d'Agrèves - 83400 Hyères - Tél. : (94) 57.20.74.

Recherche schémas complets de l'interface RS232 du TI-99, ainsi que son mode d'emploi.

Jean-Jacques Graff - 23, rue du Ballon d'Alsace - 68740 Fessenheim - Tél. : (89) 48.61.37.



# EXCLUSIF

## Plus besoin du boîtier d'extension périphérique TEXAS

Connectez directement sur votre TI/994A un contrôleur de disquettes pouvant recevoir 2 lecteurs de disquettes D.D./D.F. 5"1/4 360 ko. Permet la lecture de toutes les disquettes des programmes Texas existants.

L'ensemble comprenant le contrôleur et une unité de disquettes 4.500 F □ L'unité de disquette supplémentaire 360 Ko 2.500 F □ (garantie 1 an pièces et main-d'œuvre)

Coffret pouvant contenir 2 unités de disquettes

Interface pour imprimante RS 232 ou parallèle



Unité centrale TI 99/4A

Module Logo 2

Contrôleur de disquettes

Mémoire 32 k

### MINI-MÉMOIRE

Ce module ne se contente pas de vous offrir 4 Ko de mémoire RAM (alimentée par pile) permettant la sauvegarde des programmes et données lorsque vous éteignez l'ordinateur...

Il a aussi :

- 4 Ko de mémoire morte (ROM) et 6 Ko de mémoire morte graphique (G - ROM) contenant des utilitaires très intéressants :
    - accès possible à l'extension 32 K en TI basic ;
    - chargement de programmes-objets écrits en assembleur, soit sur mini-mémoire, soit sur extension 32 K ;
    - utilisation de PEEK et POKE, appel de sous-programmes habituellement non accessibles ;
    - programme de recherche d'erreurs (Easy Bug - Debug).
- Le module avec manuel + manuel assembleur sur mini-mémoire ..... 895 F □

### MÉMOIRE 32 K

Fonctionne avec le basic étendu.

Elle est indispensable pour la programmation en logo. Si l'on possède la mini-mémoire ou l'assembleur, elle permet d'adresser des sous-programmes en assembleur et de les exécuter.

Le module ..... 1 340 F □

### LOGO 2 (module)

Enseignez à votre ordinateur : formes, couleurs, musique, procédures, caractères dessinés, variables de tous genres, constructions grammaticales et arithmétiques...

le module logo ..... 895 F □

### MODULE SUPERGRAPH

Un nouveau basic étendu avec 35 fonctions supplémentaires permettant de tracer lignes, cercles, ellipses, axes, diagrammes en barres ou circulaires. Ce basic graphique présente 2 caractéristiques très utiles : une copie d'écran, graphiques et textes (codes ASCII), Vpoke et Vpeek accès direct à la Ram de contrôle écran ..... 1 200 F □

### BASIC ÉTENDU

Module comprenant un langage de programmation renforçant le Basic du TI 99. 40 commandes supplémentaires, accès au langage assembleur. (Entrées/sorties). Sous-programmes, stockage, sprites, traitement d'erreurs. Expressions logiques. LET multiple. Introductions multiples. Accès à l'extension 32 K. 800 F □

### LOT N° 1 INDISPENSABLE

- Module BASIC ÉTENDU manuel en français
- K7 Basic par soi-même
- K7 Lunar Lander 2
- K7 Introduction aux jeux sur TI 99 n° 1 ..... 850 F □

### LOT N° X

Module supergraph + Mémoire 32 K externe ..... 2 390 F □

### LOT N° Z TOUT LE LOGO

Module logo + mémoire 32 K ..... 1 995 F □

### BON DE COMMANDE - 99 M/7

Nom .....  
 Prenom .....  
 Adresse .....  
 Tel. ....  
 Code Postal .....  
 Ville .....

Ces prix sont indicatifs et peuvent être modifiés sans préavis. Produits disponibles dans la limite de nos stocks en magasin. Participation aux frais de port et d'expédition en recommande express pour les logiciels : + 30 F. Se renseigner pour les colis au-dessus de 5 kg ainsi que pour les expéditions à l'étranger.

LA RÈGLE À CALCUL : 65/67, bd Saint-Germain, 75005 PARIS

Tel. : 325.68.88 - Telex : ETRAV 220064 F / 1303 RAC.

Livraison des produits disponibles sous 8 jours. Parking gratuit Maubert-Lagrange.